A Technical Primer on Marginalization for Nonlinear Least Squares on Differentiable Manifolds

Evan Levine
Microsoft

February 13, 2023

1 Abstract

Many estimation problems in robotics can be cast as nonlinear least squares problems on graphs. In a large subset, it is desirable to solve for a subset of variables while eliminating the nuisance variables. This is critical when estimation is performed in a recursive, incremental fashion where the problem is augmented with new variables and observations after a solution is required. The common approach is to make linear approximations of residuals involving variables to eliminate, which allows analytically marginalizing out the variables to eliminate them. The operation is often simply called "marginalization" in the literature. It has been applied extensively, and excellent descriptions of the computations involved can be found in literature [1, 2, 3]. However, current descriptions are limited in a few respects. None explicitly describe generic treatment of manifolds; not all discuss some implementation details including robust loss functions; most are implemented using the Schur complement rather than the more numerically stable QR factorization.

Here, we bridge these gaps with a novel and general mathematical description. We define abstractions for encapsulating the structure of manifolds, discuss implementations based on the QR factorization and Schur complement, and note other considerations. The elimination procedure is encapsulated in a generic software implementation that we make available to the community, enabling marginalization in one line of code with the popular Ceres solver. We include experimental results with a simple, intuitive example of a mass-spring system to demonstrate the computational benefit of maginalization and in a sensor fusion problem to demonstrate the software abstractions introduced.

2 Introduction

In robotics, nonlinear least-squares on graphs has been used as an abstraction for many estimation problems. Common examples are bundle adjustment and simultaneous localization and mapping (SLAM), which involves simultaneously mapping an environment and localizing the robot. These abstractions enable a unified approach and an algorithmic arsenal from nonlinear optimization. Libraries such as Ceres [4], g2o [5], and GTSAM [6] for nonlinear least squares provide a rich language for users to specify problems while abstracting away many details in the solver. They exploit the sparse structure of the graph and use advanced optimization techniques in efficient implementations exploiting cache locality, vectorization, and various other techniques to achieve high performance.

In many contexts, it is desirable to solve for a subset of variables at reduced computational cost while eliminating the nuisance variables. The need arises when estimation is performed in an online fashion as new observations become available and new estimates are required. As the problem becomes augmented with new variables such as landmarks and state estimates, some variables must be removed to keep the problem size bounded. A prime example is visual odometry which adds and eliminates variables to maintain a sliding window of recent states [7, 8, 9, 10, 11]. Previous work proposes strategies for eliminating or marginalizing out variables [1, 3, 2], all of which require making linear approximations and committing to linearization points.

One practical challenge is that marginalization is highly complex and error-prone to work out analytically, implement, and test. Generic implementations that would enable efficient prototyping of algorithms are currently unavailable. However, a generic formulation and implementation would overcome both of these challenges by abstracting away error-prone operations and enabling testing in simple special cases such as linear least squares problems where the expected result can be characterized.

Our aim is to provide a more general description of marginalization and a generic implementation. This is achieved through the use of abstractions that encapsulate the structure of manifolds based on the $\boxplus$ operator and its inverse. We further describe implementation via QR factorization or the Schur complement, noting considerations related to robust loss functions. Experiments are used to demonstrate the impact of the method.

3 Preliminaries

3.1 $\boxplus$-Manifolds

Manifolds are mathematical sets that have smooth geometry. They provide a strong mathematical abstraction for variables of interest in many estimation problems in robotics. We consider a general class of manifolds called $\boxplus$-manifolds that provide sufficient generality for most applications. Example include Euclidean spaces, rotations in 3D, and rigid transformations. For a detailed description, see [12]. To make this report self-contained, the definitions are briefly restated here.

A $\boxplus$-manifold is a tuple $(\mathcal{S}, V, \boxplus, \boxminus)$, where $\mathcal{S} \subset \mathbb{R}^s$, $V \subset \mathbb{R}^n$ is an open set containing 0, and the operators are

$$\boxplus : \mathcal{S} \times \mathbb{R}^n \to \mathcal{S} \tag{1}$$
$$\boxminus : \mathcal{S} \times \mathcal{S} \to \mathbb{R}^n. \tag{2}$$

These operators must be differentiable on the manifold and also satisfy the following properties

- $x \boxplus 0 = x$
- $x \boxplus (y \boxminus x) = y$.
- For all $\delta \in V$, $(x \boxplus \delta) \boxminus x = \delta$
- For all $\delta_1, \delta_2 \in \mathbb{R}^n$ $|(x \boxplus \delta_1) \boxminus (x \boxplus \delta_2)| \leq |\delta_1 - \delta_2|$.

As implied by the notation, $\boxplus$ and $\boxminus$ generalize the "+" and "-" operators in Euclidean space. Note that the manifold is represented as a vector in a Euclidean space, which is suitable for storage in software as an array of floating-point numbers.

An example is $SO(2)$, the group of rotations in the 2D plane. For $x \in \mathrm{SO}(2) \subset \mathbb{R}^2$, the $\boxplus$ operator is defined as

$$x \boxplus \theta = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} x \tag{3}$$

The vector x is associated with the rotation matrix

$$\begin{bmatrix} x_1 & -x_2 \\ x_2 & x_1 \end{bmatrix}. \tag{4}$$

Note that this class of manifolds is broader than Lie groups. It includes manifolds for which $x \boxplus \delta$ is not continuous in x. A common example that arises in sensor fusion is S^2, the unit sphere in $\mathbb{R}^3$, which can represent gravity vectors with known magnitude.

3.2 Nonlinear Least Squares on Manifolds

Consider the constrained optimization problem

$$\min_x \quad C(x) \tag{5}$$

$$\text{subject to} \quad x_i \in \mathcal{M}_i, i = 1, \ldots, n, \tag{6}$$

where x consists of variables $\{x_i\}_{i=1}^{\ldots,n}$ and $\mathcal{M}_i$ is a $\boxplus$-manifold.

In nonlinear least squares, the objective takes the form

$$C(x) = \frac{1}{2} \sum_i \|f_i(x_{i_1}, \ldots, x_{i_k})\|^2, \tag{7}$$

where $f_i(\cdot)$, called the residual function, depends on the parameter blocks $x_{i_1}, \ldots, x_{i_k}$. The constraints must satisfy some additional properties described subsequently.

Robust loss functions can also be considered, but they are omitted for clarity in this section. The topic is revisited in Section 5.3.

Equation 7 is the negative log of a likelihood function, which can be written as a product of factors, each corresponding to one term in 7.

$$p(x) = \frac{1}{Z} \prod_i \exp(-\frac{1}{2}\|f_i(x_{i_1}, \ldots, x_{i_k})\|^2), \tag{8}$$

where Z is a normalizing constant.

4 Eliminating Variables

4.1 Problem setup

We aim to solve for a subset of variables excluding the variables to be eliminated, which we denote x_m. This is equivalent to approximating the marginal that would be defined by "integrating out" x_m in Equation 8 or equivalently minimizing it out in Equation 7. We can partition the variables into x_m, the variables related to them by error terms (their Markov blanket), denoted x_b, and the remaining variables x_r:

$$x = (x_m, x_b, x_r). \tag{9}$$

Let ∂x_m be the index set of all error terms involving x_m. Without loss of generality, the terms can be ordered so that $\partial x_m = \{1, 2, \ldots, |\partial x_m|\}$. This allows the objective to be written as a decomposition into two

$$C(x) = C_1(x_b, x_m) + C_2(x_b, x_r), \tag{10}$$

where

$$C_1(x_b, x_m) = \frac{1}{2} \sum_{i \leq |\partial x_m|} \|f_i(x_b, x_m)\|^2 \tag{11}$$

$$C_2(x_b, x_r) = \frac{1}{2} \sum_{i > |\partial x_m|} \|f_i(x_b, x_r)\|^2, \tag{12}$$

x_b, x_m, and x_r belong to a product of $\boxplus$-manifolds, which are also $\boxplus$-manifolds as proven in [12]. Since x_m is a $\boxplus$-manifold and the $\boxplus$ operator is surjective, we can make a change of variables from x_m to $\delta x_m \in \mathbb{R}^m$:

$$x_m = \breve{x}_m \boxplus \delta x_m \tag{13}$$

With this change of variables, the objective can be rewritten as

$$C(\delta x_m, x_b, x_r) = c_1(x_b, \delta x_m) + C_2(x_b, x_r), \tag{14}$$

where

$$c_1(x_b, \delta x_m) = C_1(x_b, \breve{x}_m \boxplus \delta x_m) \tag{15}$$

Assume that the residuals f_i are differentiable at the linearization point $\breve{x}_m, \breve{x}_b$. This allows making the following linear approximation for the first term in Equation 10.

$$\begin{aligned} c_1(x_b, \delta x_m) &= \frac{1}{2} \sum_{i \leq |\partial x_m|} \|f_i(x_b, \breve{x}_m \boxplus \delta x_m)\|^2 && (16) \\ &\approx \frac{1}{2} \sum_{i \leq |\partial x_m|} \|f_i(\breve{x}_b, \breve{x}_m) + J_{b,i}(x_b \boxminus \breve{x}_b) + J_{m,i}\delta x_m\|^2, && (17) \\ &\triangleq \tilde{c}_1(x_b, \delta x_m), && (18) \end{aligned}$$

where $\delta x_m \in \mathbb{R}^m$ is an increment in the tangent space $x_m = \breve{x}_m \boxplus \delta x_m$ and x_b respectively, and notation on f_i is overloaded with different arguments for brevity, and the Jacobians are

$$\begin{aligned} J_{m,i} &\triangleq \frac{\partial f_i}{\partial x_b}(\breve{x}_b, \breve{x}_m)\frac{\partial x_b}{\partial \delta x_b}(\breve{x}_b) && (19) \\ J_{b,i} &\triangleq \frac{\partial f_i}{\partial x_m}(\breve{x}_b, \breve{x}_m)\frac{\partial x_m}{\partial \delta x_m}(\breve{x}_m) && (20) \\ & && (21) \end{aligned}$$

Define the stacked Jacobians and residuals

$$\begin{aligned} J_m &\triangleq \begin{bmatrix} J_{m,1}^T & J_{m,2}^T & \ldots & J_{m,|\partial x_m|}^T \end{bmatrix}^T && (22) \\ J_b &\triangleq \begin{bmatrix} J_{b,1}^T & J_{b,2}^T & \ldots & J_{b,|\partial x_m|}^T \end{bmatrix}^T && (23) \\ f &\triangleq \begin{bmatrix} f_1(\breve{x}_b, \breve{x}_m)^T & f_2(\breve{x}_b, \breve{x}_m)^T & \ldots & f_{|\partial x_m|}(\breve{x}_b, \breve{x}_m)^T \end{bmatrix}^T. && (24) \end{aligned}$$

4.2 Marginalization via the Schur complement

$\tilde{c}_1$ can be expanded as

$$\tilde{c}_1(x_b, \delta x_m) = \begin{bmatrix} g_b \\ g_m \end{bmatrix}^T \begin{bmatrix} x_b \boxminus \breve{x}_b \\ \delta x_m \end{bmatrix} + \frac{1}{2} \begin{bmatrix} x_b \boxminus \breve{x}_b \\ \delta x_m \end{bmatrix}^T \begin{bmatrix} \Lambda_{bb} & \Lambda_{mb}^T \\ \Lambda_{mb} & \Lambda_{mm} \end{bmatrix} \begin{bmatrix} x_b \boxminus \breve{x}_b \\ \delta x_m \end{bmatrix} + \frac{1}{2}\|f\|^2, \tag{25}$$

where

$$g_m = J_m^T f \tag{26}$$

$$g_b = J_b^T f \tag{27}$$

$$\Lambda_{mb} = J_m^T J_b \tag{28}$$

$$\Lambda_{bb} = J_b^T J_b \tag{29}$$

$$\Lambda_{mm} = J_m^T J_m \tag{30}$$

Equation 25 is a quadratic approximation of the objective, which can be analytically minimized with respect to δx_m, yielding

$$\delta x_m^* \triangleq \arg\min_{\delta x_m} \tilde{c}_1(x_b, \delta x_m) \tag{31}$$

$$= -\Lambda_{mm}^\dagger (g_{mm} + \Lambda_{mb}(x_b \boxminus \breve{x}_b)), \tag{32}$$

where $\Lambda_{mm}^\dagger$ is the pseudoinverse of Λ_{mm}.

Assume that the result of this minimization $\breve{x}_m \boxplus \delta x_m^*$ is also feasible. Substituting this into $\tilde{c}_1(\cdot,\cdot)$ yields

$$\tilde{c}_1(x_b, \delta x_m^*) = g_t^T (x_b \boxminus \breve{x}_b) + \frac{1}{2}(x_b \boxminus \breve{x}_b)^T \Lambda_t (x_b \boxminus \breve{x}_b) + \frac{1}{2}\|f\|^2, \tag{33}$$

where

$$\Lambda_t = \Lambda_{bb} - \Lambda_{bm}\Lambda_{mm}^\dagger \Lambda_{bm}^T \tag{34}$$

$$g_t = g_b - \Lambda_{bm}\Lambda_{mm}^\dagger g_m. \tag{35}$$

Equation 34 is the *Schur complement* of Λ_{mm} of Λ.

Λ_t can be singular, which is problematic for some optimization methods. An eigen-decomposition of Λ_t is used to address this problem [1]. By the spectral theorem in linear algebra, Λ_t has an orthonormal eigenbasis. Let r be the rank of Λ_t and the eigen-decomposition of Λ_t be

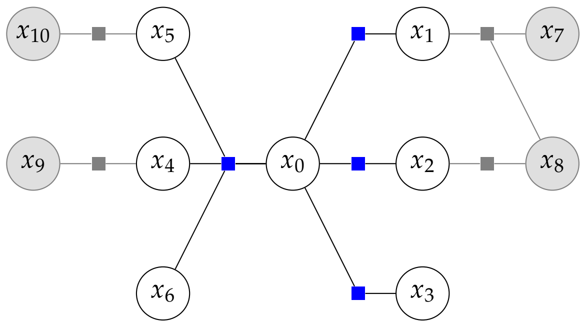
$$\Lambda_t = UDU^T, \tag{36}$$

Shown in Appendix A, Equation 4.3 can be converted to the form of a nonlinear least-squares problem given U and D from Λ_t as well as g_t:

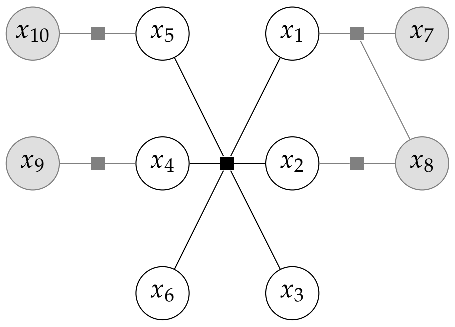
$$\tilde{c}_1(x_b, \delta x_m^*) = \frac{1}{2}\|D^{1/2}U^T(x_b \boxminus \breve{x}_b) + D^{-1/2}U^T g_t\|^2. \tag{37}$$

If it is known that Λ_t is full-rank, there is a computationally cheaper alternative, which is to compute a Cholesky factorization of Λ_t which allows computing a matrix S such that $SS^T = \Lambda_t$. In this case, the cost function becomes

$$\tilde{c}_1(x_b, \delta x_m^*) = \frac{1}{2}\|S^T(x_b \boxminus \breve{x}_b) + (S^T)^{-1} g_t\|^2. \tag{38}$$



(a) Example factor graph with variables to marginalize $x_m = (x_0)$, their Markov blanket $x_b = (x_1, x_2, \dots, x_6)$, and remaining variables $x_r = (x_7, \dots, x_{10})$



(b) Factor graph after eliminating x_0. The marginalized variable and all factors that involve it are replaced with a single factor connected to the Markov blanket.

Figure 1: Elimination of variables is local in the graph, affecting the variables to marginalize and ones related to them by error terms, their Markov blanket.

Relating the result to the original objective, one can summarize the steps in eliminating x_m as follows

$$\min_{x_m} C(x) = C_2(x_b, x_r) + \min_{\delta x_m} c_1(x_b, \delta x_m) \tag{39}$$

$$\approx C_2(x_b, x_r) + \min_{\delta x_m} \tilde{c}_1(x_b, \delta x_m) \tag{40}$$

$$= C_2(x_b, x_r) + \tilde{c}_1(x_b, \Lambda_{mm}^{\dagger}(g_{mm} + \Lambda_{mb}(x_b \boxminus \breve{x}_b))), \tag{41}$$

A graphical view of the method is shown in Figure 1. Note that the new objective can be re-linearized with respect to x_b while it has the linearization point $\breve{x}_b$ baked into it– its parameters are functions of $\breve{x}_b$ and $\breve{x}_m$. A graphical view of the cost function for the marginalization prior is shown in Figure 2.

4.3 Marginalization via QR factorization

Marginalization can be performed on the QR factorization of the Jacobian, which is algebraically equivalent to the Schur complement. We analyze this procedure with a slightly novel proof using basic properties of the pseudoinverse. The linearization in Equation 17 can be written compactly as

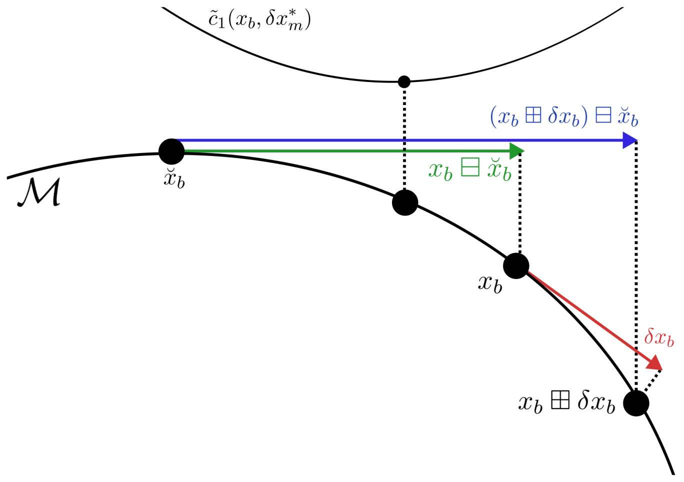
$$\tilde{c}_1(x_b, \delta x_m) = \frac{1}{2} \sum_{i \in \partial x_m} \| f_i(x_b, \breve{x}_m \boxplus \delta x_m) \|^2 \tag{42}$$

$$\approx \frac{1}{2} \| f + \begin{bmatrix} J_m & J_b \end{bmatrix} \begin{bmatrix} \delta x_m \\ (x_b \boxminus \breve{x}_b) \end{bmatrix} \|^2. \tag{43}$$

Here, we do not assume that J_b and J_m are full-rank. Let r be the rank of J_m, so that $r \le m$, and let N be the dimension of the residual f. The specialized QR factorization in [2] of the Jacobian provides the following factorization of the Jacobian.

$$\begin{bmatrix} J_m & J_b \end{bmatrix} = QR, \tag{44}$$

$$R = \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}, \tag{45}$$



$\tilde{c}_1(x_b, \delta x_m^*)$
$(x_b \boxplus \delta x_b) \boxminus \breve{x}_b$
$x_b \boxminus \breve{x}_b$
$\mathcal{M}$
$\breve{x}_b$
x_b
δx_b
$x_b \boxplus \delta x_b$

Figure 2: The cost associated with the marginalization prior $\tilde{c}_1$ is a quadratic function of the deviation in the Markov blanket from a linearization point $x_b \boxminus \breve{x}_b$. This deviation is represented in the tangent space to the manifold $\mathcal{M}$ at $\breve{x}_b$, superimposed below the cost function. The minimizer of the marginalization prior term shown in yellow may deviate from the linearization point in general. Subsequent optimizations compute the step δx_b in the tangent space at the future iterate x_b.

where Q is an orthogonal matrix, R is upper triangular, $R_{11} \in \mathbb{R}^{r \times m}$, $R_{22} \in \mathbb{R}^{N-r \times b}$, and $R_{12} \in \mathbb{R}^{r \times b}$.

$$\begin{aligned}
\tilde{c}_1(x_b, \delta x_m) &= \frac{1}{2} \| f + QR \begin{bmatrix} \delta x_m \\ (x_b \boxminus \breve{x}_b) \end{bmatrix} \|^2 && (46) \\
&= \frac{1}{2} \| Q^T f + \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix} \begin{bmatrix} \delta x_m \\ (x_b \boxminus \breve{x}_b) \end{bmatrix} \|^2 && (47) \\
&= \frac{1}{2} \| Q^T f + \begin{bmatrix} R_{11} \\ 0 \end{bmatrix} \delta x_m + \begin{bmatrix} R_{12} \\ R_{22} \end{bmatrix} (x_b \boxminus \breve{x}_b) \|^2 && (48) \\
&= \frac{1}{2} \| Q^T f + \begin{bmatrix} R_{11} \\ 0 \end{bmatrix} \delta x_m + \begin{bmatrix} R_{12} \\ R_{22} \end{bmatrix} (x_b \boxminus \breve{x}_b) \|^2 && (49)
\end{aligned}$$

Define the modified residual

$$f' \triangleq Q^T f = \begin{bmatrix} f_1' \\ f_2' \end{bmatrix}, \tag{50}$$

where $f_1 \in \mathbb{R}^r$ and $f_2 \in \mathbb{R}^{N-r}$.

The pseudoinverse provides a solution for δx_m that minimizes $\tilde{c}_1$.

$$\begin{aligned}
\delta x_m^* &= \arg\min_{\delta x_m} \tilde{c}_1(x_b, \delta x_m) && (51) \\
&= -R_{11}^{\dagger}(f_1' + R_{12}(x_b \boxminus \breve{x}_b)). && (52)
\end{aligned}$$

Substituting this into Equation 49 and some rearrangement yields

$$\tilde{c}_1(x_b, \delta x_m^*) = \frac{1}{2}\| \begin{bmatrix} f_1' \\ f_2' \end{bmatrix} - \begin{bmatrix} R_{11}R_{11}^{\dagger}(f_1' + R_{12}(x_b \boxplus \check{x}_b)) \\ 0 \end{bmatrix} + \begin{bmatrix} R_{12} \\ R_{22} \end{bmatrix} (x_b \boxminus \check{x}_b)\|^2 \tag{53}$$

$$= \frac{1}{2}\| \begin{bmatrix} f_1' \\ f_2' \end{bmatrix} - \begin{bmatrix} f_1' + R_{12}(x_b \boxplus \check{x}_b) \\ 0 \end{bmatrix} + \begin{bmatrix} R_{12} \\ R_{22} \end{bmatrix} (x_b \boxminus \check{x}_b)\|^2 \tag{54}$$

$$= \frac{1}{2}\| \begin{bmatrix} 0 \\ f_2' \end{bmatrix} + \begin{bmatrix} 0 \\ R_{22} \end{bmatrix} (x_b \boxminus \check{x}_b)\|^2 \tag{55}$$

$$= \frac{1}{2}\|f_2' + R_{22}(x_b \boxminus \check{x}_b)\|^2. \tag{56}$$

In Equation 54. We used the fact that R_{11} is has linearly independent rows, so $R_{11}^{\dagger}$ satisfies $R_{11}R_{11}^{\dagger} = I$.

The result in Equation 56 is associated with a QR factorization where $R = R_{22}$ and with the new residual f_2'. Thus, the elimination of δx_m in the QR factorization can be performed by slicing the matrix R. The result is consistent with the result in [2], but the proof is based on the pseudoinverse of R_{11} rather than the SVD of the corresponding Jacobian block. In addition, [2] proves that the cost is the same as in .

5 Additional implementation details

5.1 Automatic reparameterization

When x_m and x_b consist of poses, there is freedom in choosing the coordinate system on which they are based. As discussed in related work, the global coordinate frame can be a poor choice because a stable global-frame linearization point may not exist [1, 3]. Thus, prior to marginalization, it can be advantageous to re-represent any poses expressed in the global frame in a local and stable coordinate frame. To complement the previous description of marginalization, we provide an alternative generic description of the procedure described in previous work.

Let G be a group and $x_{wk} \in G$ be the original global-frame representation for node k in the elimination clique. Choosing node 1 as the node defining the new local coordinate system, the rebased variables are defined as

$$\begin{bmatrix} x_{w1} \\ x_{12} \\ x_{13} \\ \vdots \end{bmatrix} \triangleq r(\begin{bmatrix} x_{w1} \\ x_{w2} \\ x_{w3} \\ \vdots \end{bmatrix}) = \begin{bmatrix} x_{w1} \\ x_{w1}^{-1} \circ x_{w2} \\ x_{w1}^{-1} \circ x_{w3} \\ \vdots \end{bmatrix} \tag{57}$$

Consider a residual term in the elimination clique f, which may already be a function of x_{wk} for all $k \geq 1$. As part of the rebasing operation, f is replaced with the residual

$$f'(x_{w1}, x_{12}, x_{13}, \ldots) \triangleq f(x_{w1}, x_{w1} \circ x_{12}, x_{w1} \circ x_{13}, \ldots) \tag{58}$$

Applying the chain rule, the derivatives of f' with respect to the rebased variables are

$$\frac{\partial f'}{\partial x_{w1}} = \frac{\partial f}{\partial x_{w1}} + \sum_{k \geq 2} \frac{\partial f}{\partial x_{wk}} \frac{\partial}{\partial x_{w1}}(x_{w1} \circ x_{1k}) \tag{59}$$

$$\frac{\partial f'}{\partial x_{1k}} = \frac{\partial f}{\partial x_{wk}} \frac{\partial}{\partial x_{1k}}(x_{w1} \circ x_{1k}) \tag{60}$$

The result expression shows that the derivatives of the new residual are functions of the derivatives of the original residual and the derivatives of the composition operator for the group G. Thus, one can implement the modified residual without knowing any additional information about the original residual. The only group properties that must be implemented to perform this step are only the composition operators, their derivatives, and composition with the inverse of x_{w1}.

5.2 Choice of linearization points

There is a distinction made between “local" and “global" linearization points. The former are minimizers of Equation 11, and the latter are the minimizers of the original problem in Equation 7. In [3], Eckenhoff et al. use local linearization points which yield $g_t = 0$, allowing simplifications of the cost function for the marginalization prior. When the terms involving g_t are included, superior results can be obtained with global linearization points [13]. Both works describe advantages to relative-frame linearization points over global-frame linearization points. An additional consideration is statistical consistency, which can require alternative choices of linearization points for computing "first-estimates Jacobians" [14, 8, 15].

5.3 Outliers and Robust Loss Functions

Many relevant applications must cope with outlier measurements (e.g. feature reprojections). This is commonly addressed by augmenting the cost in Equation 7 with robust loss functions. The modified objective can be written as

$$C_{\text{robust}}(x) = \frac{1}{2}\sum_i \rho_i(\|f_i(x_{i_1}, \ldots, x_{i_k})\|^2), \tag{61}$$

where ρ_i is a robust loss function, such as the Huber loss. The quadratic approximation in Equation 25 must be modified to account for the high degree of nonlinearity introduced. Gauss-Newton optimization addresses the same problem in computing quadratic approximations of the objective. A common approach is described in [16]. Following this approach, the residuals and Jacobians are weighted before the Schur complement, resulting in a quadratic objective.

5.4 Constraints

The analytical minimization of δx_m in Equation 31 requires dropping constraints on x_m at the time of marginalization. In addition, any constraints on the Markov blanket are not considered in the linearization in Equation 17.

6 Example

The potential energy of a (possibly nonlinear) spring with constant k, endpoints at positions x_i and x_j, and equilibrium point z is

$$P_{ij} = \frac{1}{2}k\|z - x_i + x_j\|^p, \tag{62}$$

Ideal, linear springs satisfy this equation for $p = 2$.

We can describe the potential energy of a system of masses connected by both linear and nonlinear springs. The system is described as a graph where the vertices are masses and the edges are springs. Let $\mathcal{E}$ be the set of edges containing pairs of indices corresponding to masses. The total potential energy of the system is

$$P = \frac{1}{2} \sum_{(i,j)\in\mathcal{E}} k_{ij} \|z_{ij} - x_i + x_j\|^{p_{ij}}. \tag{63}$$

We can compute the minimum-energy solution by minimizing P with respect to the positions of the masses, a nonlinear least squares problem.

As a specific example, consider a 5×5 grid of masses, where each mass is connected to the ones above, below, left, and right of it by springs. Suppose that only the springs connected to the masses at the corners are nonlinear with $p = 4$ in Equation 62. Only 3 masses in each of the 4 corners are connected to nonlinear springs, and let this set be denoted $\mathcal{N}$.

The problem is a nonlinear least squares problem, but most residuals are linear. How can we exploit the sparsity of nonlinear constraints to accelerate optimization? The efficient algorithm is 1) marginalize out the mass positions except for those in $\mathcal{N}$, 2) solve for the masses in $\mathcal{N}$, 3) replace the marginalization prior with the marginalized variables and constraints, and 4) solve for the remaining variables with the ones in $\mathcal{N}$ fixed. Step 4 is a linear least squares problem, and thus, it can be solved efficiently (e.g. with a sparse Cholesky factorization followed by back-substitution). The original problem is solved because the linear approximation in Equation 17 is exact. We refer to the algorithm as "two-stage optimization with marginalization."

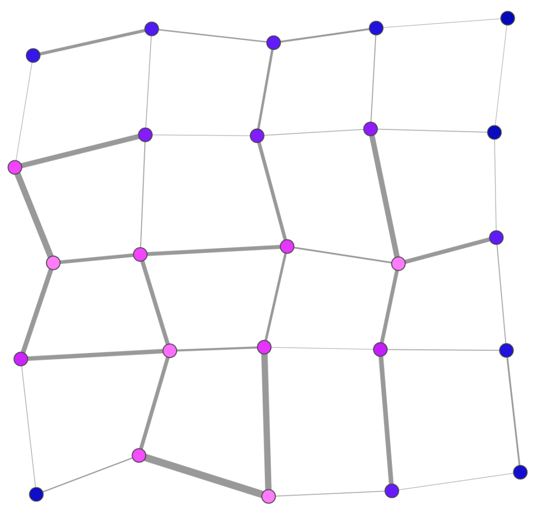
Table 1 shows the runtimes for the two-stage optimization to the runtime for a vanilla Gauss-Newton optimization on the entire state. In both algorithms, Cholesky factorization was used as the linear solver. In the first stage of the two-stage optimization, a fixed number of Gauss-Newton iterations, 10, were used to solve for the positions of masses connected to nonlinear springs. In the second stage, a linear least squares problem was solved with a (dense) Cholesky factorization. In the single stage Gauss-Newton optimization, a fixed 10 iterations of Gauss-Newton were performed. The implementation was based on the Ceres solver running on a CPU (Intel Core i7, 4.0 GHz, 4 core, 16 GB RAM).

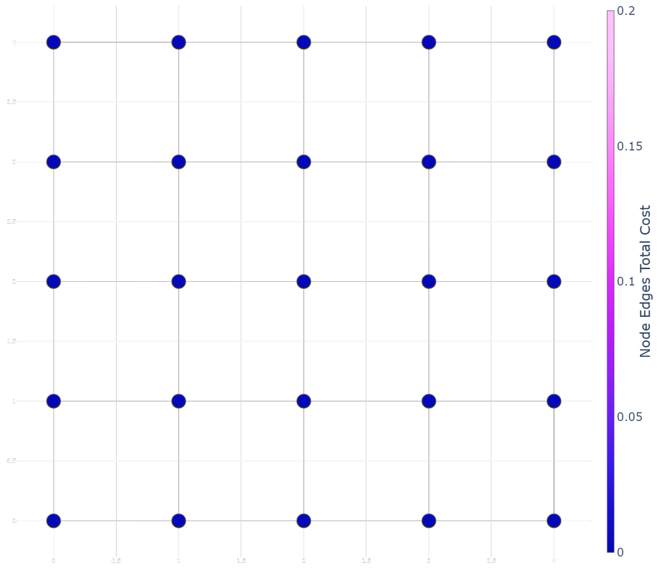
Method	Runtime
Single stage Gauss-Newton	14.2 ms
Two-stage with marginalization	1.4 ms

Table 1: In a problem where only a sparse subset of variables is involved in nonlinear residual functions, two-stage optimization procedure based on marginalization can be significantly faster. For the mass-spring system in Figure 3, the efficiency is reflected in a 10-fold reduction in runtime.

7 Acknowledgements

The author appreciates discussions with Adam Harmat, Sameer Agarwal, Tim Pfeifer, and Nikolaus Demmel.


(a) Initial state



(b) Final state

Figure 3: Initial and final states of the mass-spring system. Edges are springs with thickness scaled by their potential energy. Vertices are colored according to the total potential energy of springs connected to them.

References

[1] Nicholas Carlevaris-Bianco, Michael Kaess, and Ryan M Eustice. "Generic node removal for factor-graph SLAM". In: *IEEE Transactions on Robotics* 30.6 (2014), pp. 1371–1385.

[2] Nikolaus Demmel et al. "Square Root Marginalization for Sliding-Window Bundle Adjustment". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 13260–13268.

[3] Kevin Eckenhoff, Liam Paull, and Guoquan Huang. "Decoupled, consistent node removal and edge sparsification for graph-based SLAM". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 3275–3282.

[4] Mierle K. S. Agarwal et al. "Ceres solver". In: *http://ceres-solver.org* (2015).

[5] Giorgio Grisetti et al. "g2o: A general framework for (hyper) graph optimization". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 2011, pp. 9–13.

[6] Frank Dellaert and Chris Beall. "GTSAM 4.0". In: *URL: https://bitbucket. org/gtborg/gtsam* (2017).

[7] Gabe Sibley, Larry Matthies, and Gaurav Sukhatme. "A sliding window filter for incremental SLAM". In: *Unifying perspectives in computational and robot vision* (2008), pp. 103–112.

[8] Stefan Leutenegger et al. "Keyframe-based visual–inertial odometry using nonlinear optimization". In: *The International Journal of Robotics Research* 34.3 (2015), pp. 314–334.

[9] Jakob Engel, Vladlen Koltun, and Daniel Cremers. "Direct sparse odometry". In: *IEEE transactions on pattern analysis and machine intelligence* 40.3 (2017), pp. 611–625.

[10] Vladyslav Usenko et al. "Visual-inertial mapping with non-linear factor recovery". In: *IEEE Robotics and Automation Letters* 5.2 (2019), pp. 422–429.

[11] Tong Qin, Peiliang Li, and Shaojie Shen. "Vins-mono: A robust and versatile monocular visual-inertial state estimator". In: *IEEE Transactions on Robotics* 34.4 (2018), pp. 1004–1020.

[12] Christoph Hertzberg et al. "Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds". In: *Information Fusion* 14.1 (2013), pp. 57–77.

[13] Daniel Wilbers, Lars Rumberg, and Cyrill Stachniss. "Approximating marginalization with sparse global priors for sliding window SLAM-graphs". In: *2019 Third IEEE International Conference on Robotic Computing (IRC)*. IEEE. 2019, pp. 25–31.

[14] Guoquan P Huang, Anastasios I Mourikis, and Stergios I Roumeliotis. "A first-estimates Jacobian EKF for improving SLAM consistency". In: *Experimental Robotics*. Springer. 2009, pp. 373–382.

[15] Joel A Hesch et al. "Observability-constrained vision-aided inertial navigation". In: *University of Minnesota, Dept. of Comp. Sci. & Eng., MARS Lab, Tech. Rep* 1 (2012), p. 6.

[16] Bill Triggs et al. "Bundle adjustment—a modern synthesis". In: *International workshop on vision algorithms*. Springer. 1999, pp. 298–372.

A Cost function from the Schur complement

Here, we prove equality of the conversion from the cost in Equation 4.3 and the sum-of-squares cost in Equation 37.

Let $\delta x_b = x_b \boxminus \check{x}_b$. Substituting Equations 26 - 30 into Equation 4.3 yields

$$\tilde{c}_1(x_b, \delta x_m^*) = f^T P J_b \delta x_b + \frac{1}{2}\delta x_b^T J_b^T P J_b \delta x_b + \frac{1}{2}\|f\|^2, \tag{64}$$

where

$$P = I - J_m (J_m^T J_m)^{-1} J_m^T. \tag{65}$$

We can see that $\tilde{c}_1$ has the form of a nonlinear least-squares cost function. Specifically, there exists an A with linearly independent columns and b such that

$$\tilde{c}_1(x_b, \delta x_m^*) = \frac{1}{2}\delta x_b^T A^T A \delta x_b + b^T A \delta x_b + \frac{1}{2}\|b\|^2. \tag{66}$$

Consider the singular value decomposition

$$A = U_A \Sigma_A V_A^T, \tag{67}$$

where Σ_A contains the nonzero singular values on the diagonal and

$$U_A^T U_A = U_A U_A^T = I \tag{68}$$

$$V_A^T V_A = I \tag{69}$$

With access to $A^T A = V_A \Sigma_A^2 V_A^T$, we can compute Σ_A^2 and V_A. Since $g_t = A^T b$, we have

$$\tilde{c}_1(x_b, \delta x_m^*) = \frac{1}{2}\delta x_b^T V_A \Sigma U^T U \Sigma V_A^T \delta x_b + g_t^T V_A \Sigma_A^{-1} \Sigma_A V_A^T \delta x_b + \frac{1}{2} g_t^T V_A \Sigma^{-2} V_A^T g_t \tag{70}$$

The first terms in Equation 4.3 and Equation 70 can be shown to match.

$$\frac{1}{2}\delta x_b^T V_A \Sigma_A U^T U \Sigma_A V_A^T \delta x_b = \frac{1}{2}\delta x_b^T V_A \Sigma_A^2 V_A^T \delta x_b \tag{71}$$

$$= \frac{1}{2}\delta x_b^T A^T A \delta x_b \tag{72}$$

$$= \frac{1}{2}\delta x_b^T \Lambda_t \delta x_b \tag{73}$$

To show that the second term matches, we substitute $g_t = A^T b$, the SVD for A, and Equation 69

$$g_t^T V_A \Sigma_A^{-1} \Sigma_A V_A^T \delta x_b = b^T A V_A \Sigma_A^{-1} \Sigma_A V_A^T \delta x_b \tag{74}$$

$$= b^T U_A \Sigma_A V_A^T V_A \Sigma_A^{-1} \Sigma_A V_A^T \delta x_b \tag{75}$$

$$= b^T U_A \Sigma_A V_A^T \delta x_b \tag{76}$$

$$= b^T A \delta x_b \tag{77}$$

$$= g_t^T \delta x_b, \tag{78}$$

To show that the third term matches, the same substitutions can be used in addition to Equation 68.

$$\frac{1}{2} g_t^T V_A \Sigma_A^{-2} V_A^T g_t = \frac{1}{2} b^T A V_A \Sigma_A^{-2} V_A^T A^T b \tag{79}$$

$$= \frac{1}{2} b^T U_A \Sigma_A V_A^T V_A \Sigma_A^{-2} V_A^T V_A \Sigma_A U_A^T b \tag{80}$$

$$= \frac{1}{2} b^T U_A \Sigma_A \Sigma_A^{-2} \Sigma_A U_A^T b \tag{81}$$

$$= \frac{1}{2} b^T U_A U_A^T b \tag{82}$$

$$= \frac{1}{2} \|b\|^2 \tag{83}$$