New Square-root and Diagonalized Kalman Smoothers

Federico Wadehn, Lukas Bruderer, Vijay Sahdeva, Hans-Andrea Loeliger
Dept. of Information Technology and Electrical Engineering, ETH Zurich, Switzerland

***Abstract*—Standard implementations of Kalman filters and smoothers often suffer from numerical instability issues, due to round-off errors, even for moderate-sized state space models. Recently, two inversion-free and computationally efficient Kalman smoothers, an adapted version of the Modified-Bryson Frasier smoother (MBF), mainly tailored to input estimation and the Backward Information Filter Forward Marginal (BIFM) smoother for state estimation and output interpolation were presented. In this paper, we will first suggest improvements to both the MBF and BIFM smoother implementations aimed at improving computational efficiency and, using this improved version of the BIFM smoother, we will elaborate on its usage in sensor networks with spatially correlated noise. The main novelty in this paper is the square-root version of the BIFM smoother, which can be used in numerically critical smoothing problems, as exemplified in a force estimation problem using a multi-mass resonator model of an industrial milling machine.**

I. Introduction

Standard implementations of Kalman filters and smoothers are prone to numerical instability issues. These are caused by round-off errors in finite precision arithmetic, especially on embedded hardware. Numerical stability is a limiting factor in estimation problems featuring large dynamic ranges of the states or parameters [1], and in high dimensional state space models [2]. Numerical instability frequently originates in matrix inversions, e.g. the Rauch-Tung-Striebel (RTS) smoother [3], requires an inversion of the state covariance matrix at each time step. Therefore, we will elaborate on two matrix-inversion-free Kalman smoothers: Firstly, on the Modified-Bryson-Frazier (MBF) [4] and its extension to input estimation [5] and secondly, on the recently proposed Backward Information Filter Forward Marginal (BIFM) smoother [5]. In section II-A we will propose improvements to standard implementations of the MBF and BIFM smoother. Further, we will show via numerical simulations, the superiority of the BIFM smoother compared to standard Kalman smoothers for state estimation problems in (large) sensor networks featuring spatially correlated noise.

For poorly conditioned state space models, even these improved implementations might run into numerical problems. In these systems, numerical errors often manifest themselves in loss of symmetry and more importantly loss of positive semi-definiteness of covariance matrices [6]. We will present two approaches to alleviate this problem: Firstly, Gaussian square-root message passing and secondly, a specific state space reparametrization for diagonalizing the state covariance matrix. Both approaches are aimed at reducing the condition number of the state covariance and precision matrix. The main contributions of this paper are:

1) Tabulated square-root Gaussian message passing rules for composition of both known and novel filters and smoothers, in particular:
2) a square-root version of the BIFM smoother from [5],
3) a square-root version of the MBF input estimator [7].
4) State reparametrization for increased numerical stability

Two state estimation problems are used to illustrate the superior numerical robustness of the BIFM and square-root smoothers: An object tracking problem using a sensor network with spatially correlated measurement noise, testifies to the advantage of using Kalman smoothers based on information filters, for (large scale) sensor networks. A force estimation problem from noisy dynamometer readings, is used to show the superior numerical robustness of the proposed square-root smoothers.

A. Signal Model and Factor Graphs

A given (measured) discrete-time signal $y = (y_1, \dots, y_N)$ with $y_k \in \mathbb{R}^d$, will be modeled as the output of a linear state space model (SSM):

$$\begin{aligned} X_k &= AX_{k-1} + BU_{k-1} + W_{k-1} \\ Y_k &= CX_k + Z_k, \end{aligned} \tag{1}$$

with $\mathbf{A} \in \mathbb{R}^{n\times n}$, X_k, $W_k \in \mathbb{R}^n$, $\mathbf{B} \in \mathbb{R}^{m\times n}$, $\mathbf{C} \in \mathbb{R}^{d\times n}$. The input signal $U = (U_0, \dots, U_{N-1})$ is a sequence of independent Gaussian random variables, $W = (W_0, \dots, W_{N-1})$ represents zero-mean Gaussian state noise and $Z = (Z_1, \dots, Z_N)$ zero-mean Gaussian measurement noise. Since the state trajectory forms a Markov process, the joint density has the following factorization:

$$p(y_{1:N}, x_{0:N}|u_{0:N-1}) = p(x_0)\prod_{i=1}^{N} p(y_k|x_k)p(x_k|x_{k-1}, u_{k-1}) \tag{2}$$

which can be represented using a factor graph [8] as shown in Fig. 1.

In contrast to the classical minimum mean square error (MMSE) optimization view on Kalman filtering, we will interpret Kalman filtering and its variations as inference on probabilistic graphical models, especially on factor graphs [8]. This interpretation and notation provides a unifying view on different Kalman filter/smoother implementations as well as on extensions of Kalman filtering to input estimation [9] and parameter identification [10]. Tabulated message passing rules as in [5], [8] can readily be used for composition of algorithms

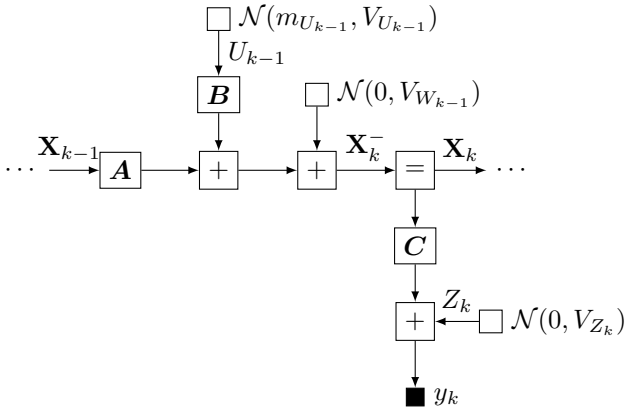


Fig. 1. Factor graph segment of the state space model. The whole factor graph consists of many such sections, one for each time step.

ranging from state estimation, to output smoothing and input estimation as shown in Section III-B.

B. Message Passing Notation

The (forward) filtering distribution $p(x_k|y_1, ..., y_k)$, e.g. computed via the covariance Kalman filter, is parametrized by the forward mean $\vec{m}_{X_k}$ and the forward covariance matrix $\vec{V}_{X_k}$. The (backward) filtering distribution $p(x_k|y_k, ..., y_N)$, e.g. computed via the backward information filter, is parametrized by the backward transformed mean $\overleftarrow{\xi}_{X_k} = \overleftarrow{W}_{X_k}\overleftarrow{m}_{X_k}$ and backward precision $\overleftarrow{W}_{X_k} = \overleftarrow{V}_{X_k}^{-1}$. Following [5], [8], right pointing arrows indicate dependence of the estimates on present and past observations, whereas left pointing arrows indicate dependence on present and future observations. The smoothing distribution $p(x_k|y_1, ..., y_N)$ is parametrized via the mean m_{X_k} and covariance V_{X_k} (BIFM smoother) or alternatively by the dual mean $\tilde{\xi}_{X_k}$ and dual precision $\tilde{W}_{X_k}$ (MBF smoother) [5].

C. Numerical Stability and Computational Complexity

The condition number, i.e. the ratio between the largest and the smallest singular value of a matrix $\kappa(V) = \sigma_{\max}(V)/\sigma_{\min}(V)$ is a key quantity for expressing bounds on numerical accuracy involving matrix inversions and multiplications [11]. Poorly conditioned covariance matrices often result in numerical instability of Kalman smoothers [12]. To increase numerical stability, a state transformation and square-root message passing, both aimed at reducing the condition number and enforcing symmetry and positive definiteness of covariance and precision matrices, are presented in section II-C and III respectively. Most modern CPUs and programming languages have native support for floating point arithmetic, hence computational efficiency of different Kalman smoothers will be assessed in terms of FLOPS [13].

II. Implementations of MBF and BIFM Smoothers

In this section we will show different methods to enhance numerical stability and reduce computational requirements of the MBF and BIFM smoother, especially in sensor networks and large scale state space models.

A. Composite Blocks and Computational Efficiency

Note from the following simple example that different implementations of mathematically equivalent expressions can result in substantial gains in computational efficiency: Let $A \in \mathbb{R}^{n\times n}$ and $v \in \mathbb{R}^n$, then the left-hand side and right-hand side of $(Av)v^{\mathsf{T}} = A(vv^{\mathsf{T}})$ are mathematically equivalent, however the left-hand side involves one matrix-vector $O(n^2)$ and one outer product $O(n^2)$, compared to the right-hand side, which involves one outer product $O(n^2)$ and one matrix-matrix multiplication $O(n^3)$. Similarly, naive composition of tabulated message update rules rarely yields the most efficient and numerically stable implementation of the desired algorithm. Therefore further algebraic manipulations should be performed to reduce computational cost and increase numerical robustness of the MBF and BIFM smoother. Combining equations [(III.8), [5]] and [(V.6), [5]] of the standard MBF smoother from [5] for propagating the dual precision matrix $\tilde{W}_{X_k}$, through the A-node and observation block (equality- and C-node) in Fig. 1 yields:

$$\tilde{W}_{X_{k-1}} = A^{\mathsf{T}}F_k^{\mathsf{T}}\tilde{W}_{X_k}F_kA + A^{\mathsf{T}}C^{\mathsf{T}}G_kCA, \tag{3}$$

where F_k and G_k (cf. Table IV) are summary quantities obtained in the Kalman filtering step. By introducing $F_k^a = F_kA$, i.e. combining the multiplier- and equality-node, we can avoid two matrix multiplications of $n \times n$ matrices, where n is the dimension of the state. One matrix multiplication can be saved in (3) and the other in the computation of:

$$F_k^a = F_kA = (I - \vec{V}_{X_k^-}C^{\mathsf{T}}G_kC)A. \tag{4}$$

For the BIFM smoother (being dual to the MBF) introducing the auxiliary variable $\tilde{F}_k^a = A^{\mathsf{T}}\tilde{F}_k$ allows combining the A-node with the input-block, which results in the same computational savings as for the MBF smoother.

B. Efficient State Estimation in Sensor Networks

Straightforward application of the tabulated Gaussian message passing rules in [5], leads to an inversion of a matrix with row and column dimension equal to the dimension of the output vector y_k for the MBF and of the input vector u_k for the BIFM smoother. This matrix inversion is often the root cause of numerical instability of Kalman smoothers for MIMO systems. In presence of state noise, a standard implementation of the BIFM would incur a matrix inversion of the size of the state, thus losing one of its main selling points, namely being matrix inversion-free.

1) Scalar updates: When measurement or state noise are uncorrelated, i.e. their covariance matrix is diagonal, matrix inversions for Kalman and information filters can be avoided via sequential scalar updates (splitting the B and C matrix) as in [14]. Figure 2 compares the computational requirements of standard and efficient (composite blocks + scalar updates) Kalman smoother implementations in terms of FLOPS for n-th order single-input single-output (SISO) systems. Scalar addition and multiplication are counted as 1 FLOP each. Due to the vast number of different implementations for arithmetic

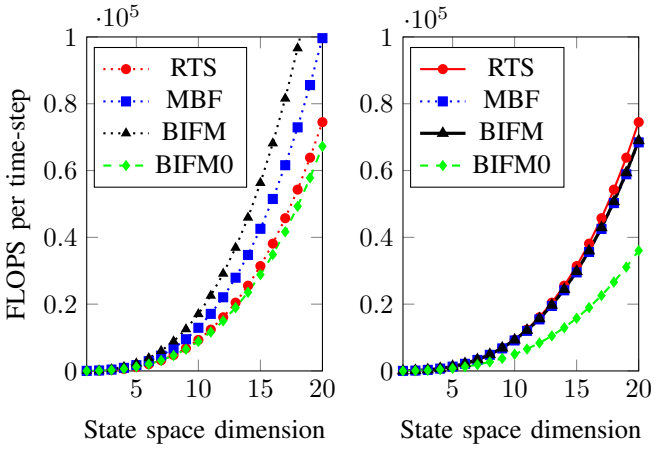


Fig. 2. Required FLOPS with standard update equations (left) and with proposed efficient update equations (right). The FLOP count for the BIFM was done both with state noise (BIFM) and without state noise (BIFM0).

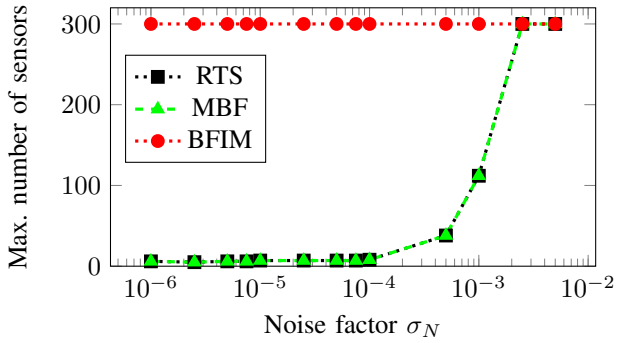


Fig. 3. Maximal number of sensors for which the relative mean squared estimation error is below 75% for the RTS, MBF and BIFM smoother as a function of the noise factor σ_Z. The BIFM smother could handle without failures the maximal number of 300 sensors set in the simulation.

operations such as matrix inversions and multiplications, we computed the FLOP count by resorting to standard textbook implementations [15]:

- Inversion of an $n \times n$ matrix: $4/3n^3 + 3/2n^2 - 5/6n$ FLOPS
- Multiplication of two $n \times n$ matrices: $2n^3 - n^2$ FLOPS
- Multiplication $n \times n$ matrix with $n \times 1$ vector: $2n^2 - n$ FLOPS
- Outer product of $n \times 1$ vectors: n^2 FLOPS
- Inner product of $n \times 1$ vectors: $2n - 1$ FLOPS

2) Spatially Correlated Measurement Noise: In sensor networks, the assumption that measurement noise is uncorrelated might not hold and severely degrade estimation accuracy if not taken into account. Different models for spatially correlated sensor noise have been proposed, among which [16] for wireless sensor networks on a 2D lattice. In this model the spatial correlation of the measurement noise is assumed to be a function of the distance between two sensors. The covariance of the noise on sensors i and j (scalar R_{ij}), decays exponentially with the Euclidean distance between the sensors:

$$R_{ij} = \sigma_Z^2 \exp(-\rho||r_i - r_j||_2), \tag{5}$$

where ρ indicates the coupling between the noise on the different sensors, $r_i, r_j \in \mathbb{R}^2$ are the positions of sensor i and j and σ_Z is a noise scaling factor.

We compared the performance of different Kalman smoother implementations using (a slightly modified version of) the state space model for target tracking from [16]. A non-maneuvering target is modeled by a continuous-time white noise acceleration model, giving rise to the discrete-time system: $A = [0.8, 0, \Delta/10, 0; 0, 0.8, 0, \Delta/10; 0, 0, 0.8, 0; 0, 0, 0, 0.8]$, $B = 0$. The state noise covariance was chosen as in [16]: $V_W = \tau[\Delta^3/3, 0, \Delta^2/2, 0; 0, \Delta^3/3, 0, \Delta^2/2; \Delta^2/2, 0, \Delta, 0; 0, \Delta^2/2, 0, \Delta]$. The first two states x_1 and x_2 denote the position and x_3 and x_4 the velocity of the target in 2D, whereas Δ, denotes the sampling time and is set equal to 0.1 and τ represents a state noise scaling factor and was set to 1. The sensors are randomly placed on a 50 by 50 lattice and the rows of the $\mathbf{C}$ matrix are independently drawn from a multivariate standard Normal distribution as in [16].

For each noise scaling factor level σ_Z in (5), the number of sensors in the simulation was increased in steps of 2, until the estimation algorithm failed in more than 25 out of 100 runs, or the preset upper bound of 300 sensors was reached. The state estimation was considered as failed if the relative mean squared error ($||x - m_X||/||x||$) was above 75%. This loose bound was chosen, since when the smoothers failed, they tended to diverge, resulting in NaNs for the mean and covariance. Figure 3 shows the maximum number of sensors for the RTS, MBF and BIFM smoother as a function of the measurement noise scaling factor σ_Z. The superior numerical stability of the BIFM smoother compared to both the MBF and the RTS smoother for sensor networks with spatially correlated noise is due to the usage of the information filter in the BIFM filtering step. In the information filter, measurements enter the state estimates additively (addition of transformed means and precisions), which is a non critical operation from a numerical point of view. Most of the RTS' and the MBF's tracking failures can be attributed to the numerically challenging measurement update equation in the (covariance) Kalman filtering step.

C. State Transformation for Increasing Numerical Stability

Several methods to enhance numerical stability of Kalman filter implementations have been proposed, such as symmetrizing the covariance matrix via $((V+V^\mathsf{T})/2)$ and the Joseph covariance update [14] to retain positive definiteness. Further ad hoc methods aimed at avoiding poorly conditioned covariance matrices include: rescaling the state and system matrices, the addition of "stabilizing noise" and setting minimal thresholds on the entries of the covariance matrices [14].

When using Kalman filters and smoothers for, e.g. input estimation or output interpolation [9], the state parametrization can be chosen freely. In [12] error propagation was shown to

be mitigated when the covariance matrix is close to diagonal. Moreover, in [17], to enhance numerical stability of solvers for the algebraic Riccati equation, scaling with an appropriately chosen constant has been suggested. We extend this idea to matrix scaling by proposing state transformations, based on *steady-state* covariance and precision matrices computed with numerically robust Riccati solvers. These transformations are aimed at both reducing the condition number and keeping the time-varying covariance and precision matrices close to diagonal.

a) State Transformation for Kalman and Information Filter: To reduce the condition number of the covariance matrix and to keep its diagonal structure as close as possible, we perform the eigenvalue decomposition of the steady-state covariance matrix[1]:

$$\vec{V}_{\mathbf{X}}^{\infty} = Q\Lambda Q^{\mathsf{T}}, \tag{6}$$

with Q being the eigenbasis and Λ a diagonal matrix containing the eigenvalues. Thereby we obtain the transformation matrix $T = \Lambda^{-\frac{1}{4}} Q^{\mathsf{T}}$ (with $\bar{x} = Tx$), such that the covariance matrix of the transformed state $\vec{V}_{\bar{X}_k} = T\vec{V}_{X_k}T^{\mathsf{T}} \approx \Lambda^{\frac{1}{2}}$ is close to diagonal and has a reduced condition number. For the information filter the corresponding transformation matrix can be obtained by decomposing the steady-state precision matrix $\overleftarrow{W}_{\mathbf{X}}^{\infty}$ instead of the steady-state covariance matrix. Since we observe that for the RTS, MBF and BIFM smoother the filtering part is the numerically most problematic one, using a transformation targeting the filtering part is often sufficient.

b) Two-Filter Smoother: For the *Two-Filter smoother* [4], [8] we suggest a new transformation, based on simultaneously diagonalizing both the forward covariance matrix and backward precision. Given two symmetric and positive semidefinite matrices A and B there always exists an invertible matrix (cf. page 500 in [11]), such that $T^{\mathsf{T}}AT$ is diagonal and $T^{\mathsf{T}}BT$ is the identity matrix. The algorithm from [11] can readily be modified to find a transformation T, which simultaneously diagonalizes the steady-state (forward) covariance $\vec{V}_X^{\infty}$ and the steady-state (backwards) precision $\overleftarrow{W}_X^{\infty}$ by balancing the diagonal elements. The state space model can therefore be transformed into a basis in which both the (time-varying) transformed forward covariance $\vec{V}_{\bar{X}_k} = T\vec{V}_{X_k}T^{\mathsf{T}}$ and backward precision $\overleftarrow{W}_{\bar{X}_k} = T^{-\mathsf{T}}\overleftarrow{W}_{X_k}T^{-1}$ are close to diagonal, have balanced eigenvalues, and therefore a reduced condition number.

III. Gaussian Square-Root Message Passing

The covariance matrix V_X and the precision matrix W_X are symmetric and positive (semi-) definite. Hence, they posses a Cholesky decomposition $V_X = N_X^{\mathsf{T}}N_X$ and $W_X = S_X^{\mathsf{T}}S_X$, where N_X and S_X are upper triangular matrices [11]. Instead of propagating the covariance or the precision matrix through the nodes of the factor graph in Fig. 1, square-root filtering and smoothing algorithms propagate the Cholesky factors N_{X_k} and S_{X_k}. In the final step, these factors are combined at each node to obtain (symmetric and positive semidefinite) covariance V_{X_k} and precision matrices W_{X_k} [6].

Numerical stability is increased by square-root Kalman filtering and smoothing for two reasons: firstly, the propagated square-root quantities have a reduced dynamic range, and secondly, positive definiteness and symmetry is enforced by computing the covariance or precision matrices from their *square-roots*.

Let us now consider Gaussian square-root message passing through each of the nodes and blocks (composite nodes) in the factor graph in Fig. 1, as shown in Tables I-VI. For the square-root Kalman filter, the update equation for the mean is the same as in regular Kalman filter. For the information filter however, there are two versions: one, where the transformed mean is kept in the standard form $\vec{\xi}_X = \vec{W}_X\vec{m}_X$ as shown in [14] and the other which uses a factored mean $\vec{\zeta}_X = \vec{S}_X\vec{m}_X$. The latter version is novel to the best of the authors' knowledge and has the advantage that only square-root quantities are propagated. These have a reduced dynamic range, which makes this type of information filter numerically even more robust. This however, comes at the cost of additional QR-decompositions (cf. Tables I and V).

Regarding covariance or precision updates, there are two different forms of equations: $C^{\mathsf{T}}C = A^{\mathsf{T}}A + B^{\mathsf{T}}B$ (e.g. covariance update at the plus-node or precision update at equality-node) and $C^{\mathsf{T}}C = A^{\mathsf{T}}A - B^{\mathsf{T}}B$ (e.g. covariance update at the equality-node or precision update at the plus-node). Given the Cholesky factors A and B on the right-hand side of the equation, the goal is to obtain the Cholesky factor C, without computing the full matrix. For the sum, we proceed as follows:

$$C^{\mathsf{T}}C = A^{\mathsf{T}}A + B^{\mathsf{T}}B \tag{7}$$

$$(C^{\mathsf{T}}, 0)Q^{\mathsf{T}}Q \begin{pmatrix} C \\ 0 \end{pmatrix} = (A^{\mathsf{T}}, B^{\mathsf{T}}) \begin{pmatrix} A \\ B \end{pmatrix} \tag{8}$$

Where the upper triangular matrix $(C^{\mathsf{T}}, 0)^{\mathsf{T}}$ and the orthogonal matrix Q are obtained via the QR-decomposition $\mathrm{qr}(\cdot)$:

$$Q \begin{pmatrix} C \\ 0 \end{pmatrix} = \mathrm{qr} \begin{pmatrix} A \\ B \end{pmatrix}. \tag{9}$$

For the difference, the Cholesky factor can be obtained via the complex QR-decomposition by introducing the imaginary unit in front of the B matrix in (9) as shown in Table VI, or via the real hyperbolic householder transform, which was used in [18] to obtain the square-root MBF. The remaining proofs of the update rules of Tables I-VI are deferred to the appendix.

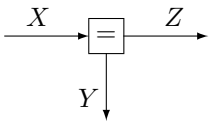
A. Kalman Filter and Smoother Implementations

In the following we will show the ease with which, using the tabulated message passing rules from Tables I-VI, both known and novel square-root Kalman filters and smoothers, as well as input estimators can be devised.

In Tables I-VI the orthogonal matrix $Q = (Q_1, Q_2)$ is obtained by QR decomposition of the matrix on the right

[1]The superscript "∞" is used here to indicate *steady-state* covariance/precision matrices.

TABLE I
GAUSSIAN SQUARE-ROOT MESSAGE PASSING THROUGH AN EQUALITY NODE.



Constraint $X = Y = Z$, expressed by $\delta(z - x)\,\delta(y - x)$. Forward precision $\overrightarrow{W} = \overrightarrow{S}^{\mathsf{T}}\overrightarrow{S}$ and covariance $V = N^{\mathsf{T}}N$.

$$\overrightarrow{\zeta}_Z = Q_1^{\mathsf{T}} \begin{pmatrix} \overrightarrow{\zeta}_X \\ \overleftarrow{\zeta}_Y \end{pmatrix} \tag{I.1}$$

$$\overrightarrow{\xi}_Z = \overrightarrow{\xi}_X + \overleftarrow{\xi}_Y \tag{I.2}$$

$$(Q_1, Q_2) \begin{pmatrix} \overrightarrow{S}_Z \\ 0 \end{pmatrix} = \mathrm{qr} \begin{pmatrix} \overrightarrow{S}_X \\ \overleftarrow{S}_Y \end{pmatrix} \tag{I.3}$$

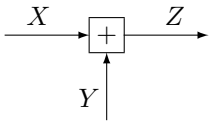
$$m_X = m_Y = m_Z \tag{I.4}$$

$$N_X = N_Y = N_Z \tag{I.5}$$

$$\tilde{\xi}_X = \tilde{\xi}_Y + \tilde{\xi}_Z \tag{I.6}$$

With the placeholder $\Psi \in \{\zeta, \xi, S\}$, for the messages in reverse direction, replace $\overrightarrow{\Psi}_Z$ with $\overleftarrow{\Psi}_X$ and $\overrightarrow{\Psi}_X$ with $\overleftarrow{\Psi}_Z$. The submatrix Q_1 represents the first n columns of Q (computed via the QR decomposition), where $n = \dim(X)$.

TABLE II
GAUSSIAN MESSAGE PASSING THROUGH AN ADDER NODE.



Constraint $Z = X + Y$, expressed by factor $\delta(z - (x + y))$. Dual precision $\tilde{W} = \tilde{S}^{\mathsf{T}}\tilde{S}$ and covariance $\overrightarrow{V} = \overrightarrow{N}^{\mathsf{T}}\overrightarrow{N}$.

$$\overrightarrow{m}_Z = \overrightarrow{m}_X + \overrightarrow{m}_Y \tag{II.1}$$

$$Q \begin{pmatrix} \overrightarrow{N}_Z \\ 0 \end{pmatrix} = \mathrm{qr} \begin{pmatrix} \overrightarrow{N}_X \\ \overrightarrow{N}_Y \end{pmatrix} \tag{II.2}$$

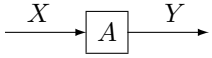
$$m_Z = m_X + m_Y \tag{II.3}$$

$$\tilde{\xi}_X = \tilde{\xi}_Y = \tilde{\xi}_Z \tag{II.4}$$

$$\tilde{S}_X = \tilde{S}_Y = \tilde{S}_Z \tag{II.5}$$

With the placeholder $\Psi \in \{m, N\}$, for the messages in reverse direction, replace $\overrightarrow{\Psi}_Z$ with $\overleftarrow{\Psi}_X$ and $\overrightarrow{\Psi}_X$ with $\overleftarrow{\Psi}_Z$ and change the sign of $\overrightarrow{m}_Y$.

TABLE III
SQUARE-ROOT GAUSSIAN MESSAGE PASSING THROUGH A MATRIX MULTIPLIER NODE WITH ARBITRARY REAL MATRIX A.



Constraint $Y = AX$, expressed by factor $\delta(y - Ax)$

$$\overrightarrow{m}_Y = A\overrightarrow{m}_X \tag{III.1}$$

$$\overrightarrow{N}_Y = \overrightarrow{N}_X A^{\mathsf{T}} \tag{III.2}$$

$$\overleftarrow{\zeta}_X = \overleftarrow{\zeta}_Y \tag{III.3}$$

$$\overleftarrow{\xi}_X = A^{\mathsf{T}}\overleftarrow{\xi}_Y \tag{III.4}$$

$$\overleftarrow{S}_X = \overleftarrow{S}_Y A \tag{III.5}$$

$$m_Y = Am_X \tag{III.6}$$

$$N_Y = N_X A^{\mathsf{T}} \tag{III.7}$$

$$\tilde{\xi}_X = A^{\mathsf{T}}\tilde{\xi}_Y \tag{III.8}$$

$$\tilde{S}_X = \tilde{S}_Y A \tag{III.9}$$

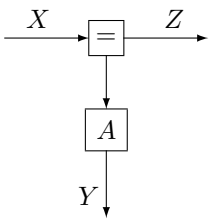
hand side of the respective covariance/precision update equation. In the corresponding update equations for the (*forward* or *backward*) transformed mean vector $\xi_X = W_X m_X$ (or $\zeta_X = S_X m_X$) and regular mean vector m_X, computations can be saved by using only the submatrices Q_1 or Q_2 (of appropriate dimensions) instead of the full Q matrix. Note that by resorting to algorithms such as the Householder QR-decomposition ([11], page 248), one can compute only the required submatrices instead of the full QR decomposition.

a) The Covariance Square-Root Filter [19]: is based on the forward recursion with the forward mean $\overrightarrow{m}_{X_k}$ and the Cholesky factor $\overrightarrow{N}_{X_k}$ of the forward covariance $\overrightarrow{V}_{X_k} = \overrightarrow{N}_{X_k}^{\mathsf{T}}\overrightarrow{N}_{X_k}$. The mean and Cholesky factor are propagated through the A-matrix using (III.1) and (III.2) respectively. The input and state noise are taken into account for, with (III.1) and (III.2) for the B-matrix, followed by (II.1) and (II.2) for the plus-node. The update step through the observation block is performed with (IV.1) and (IV.2).

b) The Information Square-Root Filter [19]: is based on the forward recursion with the forward transformed mean $\overrightarrow{\zeta}_{X_k} = \overrightarrow{S}_{X_k}\overrightarrow{m}_{X_k}$ and the Cholesky factor $\overrightarrow{S}_{X_k}$ of the forward precision $\overrightarrow{W}_{X_k} = \overrightarrow{S}_{X_k}^{\mathsf{T}}\overrightarrow{S}_{X_k}$. The transformed mean and Cholesky factor are propagated through the A-matrix using (the time-reversed version, i.e. flipping the direction of the arrows) (III.3) and (III.5) respectively. The input and state noise are accounted for with (V.1) and (V.3). The update step through the observation block is performed with (the time-reversed version of) (III.3) and (III.5) through the C-matrix followed by (I.1) and (I.3) through the equality-node.

c) The 2-Filter Smoother: is based on combining the forward mean $\overrightarrow{m}_{X_k}$ and the Cholesky factor $\overrightarrow{N}_{X_k}$ with the backward transformed mean $\overleftarrow{\zeta}_{X_k}$ and and the Cholesky factor

TABLE IV
GAUSSIAN SQUARE-ROOT MESSAGE PASSING THROUGH AN OBSERVATION BLOCK.



$$\vec{m}_Z = \vec{m}_X + \bar{R}^\mathsf{T}\tilde{G}^{-\mathsf{T}}(\overleftarrow{m}_Y - A\vec{m}_X) \quad \text{(IV.1)}$$

$$Q\begin{pmatrix} \tilde{G} & \bar{R} \\ 0 & \vec{N}_Z \end{pmatrix} = \text{qr}\begin{pmatrix} \overleftarrow{N}_Y & 0 \\ \vec{N}_X A^\mathsf{T} & \vec{N}_X \end{pmatrix} \quad \text{(IV.2)}$$

$$\tilde{\xi}_X = F^\mathsf{T}\tilde{\xi}_Z + A^\mathsf{T}G\,(A\vec{m}_X - \overleftarrow{m}_Y) \quad \text{(IV.3)}$$

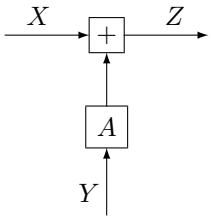
$$Q\begin{pmatrix} \tilde{S}_X \\ 0 \end{pmatrix} = \text{qr}\begin{pmatrix} \tilde{S}_Z F \\ \tilde{G}^{-\mathsf{T}} A \end{pmatrix} \quad \text{(IV.4)}$$

$$\text{with } F \triangleq I - \vec{N}_X^\mathsf{T}\vec{N}_X A^\mathsf{T} G A \quad \text{(IV.5)}$$

$$\text{and } G = \left(\tilde{G}^\mathsf{T}\tilde{G}\right)^{-1} \triangleq \left(\overleftarrow{V}_Y + A\vec{V}_X A^\mathsf{T}\right)^{-1} \quad \text{(IV.6)}$$

With $\Psi \in \{m, N\}$, for the reverse direction, replace $\vec{\Psi}_Z$ with $\overleftarrow{\Psi}_X$ and $\vec{\Psi}_X$ with $\overleftarrow{\Psi}_Z$. For $\Psi \in \{\tilde{\xi}, \tilde{S}\}$ exchange Ψ_Z and Ψ_X and change "+" to "-" in (IV.3).

TABLE V
GAUSSIAN MESSAGE PASSING THROUGH AN INPUT BLOCK.



$$\vec{\zeta}_Z = Q_2^\mathsf{T}\begin{pmatrix} 0 \\ \vec{\zeta}_X + \vec{S}_X A\vec{m}_Y \end{pmatrix} \quad \text{(V.1)}$$

$$\vec{\xi}_Z = \vec{\xi}_X + \tilde{R}^\mathsf{T}\tilde{H}^{-\mathsf{T}}\left(\vec{\xi}_Y - A^\mathsf{T}\vec{\xi}_X\right) \quad \text{(V.2)}$$

$$(Q_1, Q_2)\begin{pmatrix} \tilde{H} & \tilde{R} \\ 0 & \vec{S}_Z \end{pmatrix} = \text{qr}\begin{pmatrix} \vec{S}_Y & 0 \\ \vec{S}_X A & \vec{S}_X \end{pmatrix} \quad \text{(V.3)}$$

$$m_X = \tilde{F}^\mathsf{T} m_Z + AH\left(A^\mathsf{T}\vec{\xi}_X - \vec{\xi}_Y\right) \quad \text{(V.4)}$$

$$Q\begin{pmatrix} N_X \\ 0 \end{pmatrix} = \text{qr}\begin{pmatrix} N_Z\tilde{F} \\ \tilde{H}^{-\mathsf{T}} A^\mathsf{T} \end{pmatrix} \quad \text{(V.5)}$$

$$\text{with } \tilde{F} \triangleq I - \vec{S}_X^\mathsf{T}\vec{S}_X A H A^\mathsf{T} \quad \text{(V.6)}$$

$$\text{with } H = \left(\tilde{H}^\mathsf{T}\tilde{H}\right)^{-1} \triangleq \left(\vec{W}_Y + A^\mathsf{T}\vec{W}_X A\right)^{-1} \quad \text{(V.7)}$$

With $\Psi \in \{\zeta, \xi, S\}$, for the reverse direction, replace $\vec{\Psi}_Z$ with $\overleftarrow{\Psi}_X$ and $\vec{\Psi}_X$ with $\overleftarrow{\Psi}_Z$ and change the sign of $\vec{m}_Y$ and $\vec{\xi}_Y$. For $\Psi \in \{m, N\}$ additionally replace Ψ_Z with Ψ_X and Ψ_X with Ψ_Z. The submatrix Q_2 represents the last n columns of $Q = (Q_1, Q_2)$ (computed via the QR decomposition), where $n = \dim(X)$.

$\overleftarrow{S}_{X_k}$ of the backward precision. The *Square-Root Covariance Filter* runs forward in time, the *Square-Root Information Filter* backwards in time and (10-11) are finally used to fuse the forward and backward estimates to get the marginals m_X and $V_X = N_X^\mathsf{T} N_X$ as follows:

$$m_X = \vec{m}_X + \tilde{M}^{-\mathsf{T}}\tilde{L}^\mathsf{T}\left(\overleftarrow{\zeta}_X - \overleftarrow{S}_X\vec{m}_X\right) \quad (10)$$

$$Q\begin{pmatrix} \tilde{L} & \tilde{M} \\ 0 & N_X \end{pmatrix} = \text{qr}\begin{pmatrix} I & 0 \\ \vec{N}_X\overleftarrow{S}_X^\mathsf{T} & \vec{N}_X \end{pmatrix}. \quad (11)$$

d) The Square-Root MBF [18]: is based on the forward recursion using the *Square-Root Covariance Filter* followed by propagating the dual mean $\tilde{\xi}_{X_k} \triangleq \tilde{W}_{X_k}(\vec{m}_{X_k} - \overleftarrow{m}_{X_k})$ and the Cholesky factor of the dual precision $\tilde{S}_{X_k}$ with $\tilde{W}_{X_k} = \tilde{S}_{X_k}^\mathsf{T}\tilde{S}_{X_k}$ backwards in time. The update for the A-node is performed with (III.8 and III.9). The dual precision, as well as its square-root form are invariant to the plus-node (which makes them excellent input estimators), therefore the remaining update through the observation block is performed via (IV.3) and (IV.4). The marginal mean m_{X_k} and variance V_{X_k} are obtained via the single-edge relations (VI.7), (VI.9) and (VI.10).

e) The Square-Root BIFM: is based on the backward recursion using the *Square-Root Information Filter* and propagating the mean m_{X_k} and the Cholesky factor of the variance N_{X_k} with $V_{X_k} = N_{X_k}^\mathsf{T} N_{X_k}$ forwards. The update for the A-node is performed with (III.6) and (III.7). The update through the input and state noise block are performed via (V.4) and (V.5). This version is new to the best of the authors knowledge. Its derivation is shown in the appendix.

B. The Square-Root MBF Input Estimator

Input estimation via standard Kalman smoothing has previously been described in [9], [1]. Here we propose numerically stable square-root version of this input estimator.

The square-root MBF input estimator computes the dual mean $\tilde{\xi}_{X_k}$ and Cholesky factor $\tilde{S}_{X_k}$ of the dual precision and propagates these (backwards) through the B-node via (III.8) and (III.9) to get the likelihood of the input u_k. The (marginal) mean m_{U_k} and variance V_{U_k} of the input are finally obtained through the single-edge relations (VI.7), (VI.9) and (VI.10), which combine the likelihood (parametrized by $\tilde{\xi}_{U_k}$ and Cholesky factor $\tilde{S}_{U_k}$) with the Gaussian prior on the input (parametrized by $\vec{m}_{U_k}$ and $\vec{V}_{U_k}$).

TABLE VI
GAUSSIAN SINGLE-EDGE MARGINALS (m, V) AND THEIR DUALS $(\tilde{\xi}, \tilde{W})$.

$$\tilde{\xi}_X \triangleq \tilde{W}_X(\vec{m}_X - \overleftarrow{m}_X) \quad \text{(VI.1)}$$
$$= \vec{S}_X^{\mathsf{T}}\vec{\zeta}_X - \vec{S}_X^{\mathsf{T}}\vec{S}_X m_X \quad \text{(VI.2)}$$
$$= \overleftarrow{S}_X^{\mathsf{T}}\overleftarrow{S}_X m_X - \overleftarrow{S}_X^{\mathsf{T}}\overleftarrow{\zeta}_X \quad \text{(VI.3)}$$
$$\tilde{W}_X \triangleq (\vec{V}_X + \overleftarrow{V}_X)^{-1} = \tilde{S}_X^{\mathsf{T}}\tilde{S}_X \quad \text{(VI.4)}$$
$$Q\begin{pmatrix} \tilde{S}_X \\ 0 \end{pmatrix} = \mathrm{qr}\begin{pmatrix} \vec{S}_X \\ iN_X\vec{S}_X^{\mathsf{T}}\vec{S}_X \end{pmatrix} \quad \text{(VI.5)}$$
$$= \mathrm{qr}\begin{pmatrix} \overleftarrow{S}_X \\ iN_X\overleftarrow{S}_X^{\mathsf{T}}\overleftarrow{S}_X \end{pmatrix} \quad \text{(VI.6)}$$

$$m_X = \vec{m}_X - \vec{N}_X^{\mathsf{T}}\vec{N}_X\tilde{\xi}_X \quad \text{(VI.7)}$$
$$= \overleftarrow{m}_X + \overleftarrow{N}_X^{\mathsf{T}}\overleftarrow{N}_X\tilde{\xi}_X \quad \text{(VI.8)}$$
$$V_X = N_X^{\mathsf{T}}N_X \quad \text{(VI.9)}$$
$$Q\begin{pmatrix} N_X \\ 0 \end{pmatrix} = \mathrm{qr}\begin{pmatrix} \vec{N}_X \\ i\tilde{S}_X\vec{N}_X^{\mathsf{T}}\vec{N}_X \end{pmatrix} \quad \text{(VI.10)}$$
$$= \mathrm{qr}\begin{pmatrix} \overleftarrow{N}_X \\ i\tilde{S}_X\overleftarrow{N}_X^{\mathsf{T}}\overleftarrow{N}_X \end{pmatrix} \quad \text{(VI.11)}$$

C. Numerical Simulation of Multi-mass Resonator

In industrial milling, undesired vibrations are a major cause of machine wear and imprecision in workpiece elaboration. Therefore, monitoring (cutting) forces on the workpiece using dynamometers is essential. Apart from frequency domain approaches as in [20], in [1] and [21] model-based approaches based on Kalman filtering and smoothing were proposed.

The workpiece-sensor-machine coupling is described using the 4-mass resonator model (machine, table, sensor and workpiece) proposed in [1]. The model consists of a concatenation of second-order models, describing masses, differing significantly in weight, subject to damping and mechanical coupling. The 4-mass resonator model gives rise to an 8th order state space model or equivalently to an 8th order transfer function, whose frequency response (3.5 kHz sampling rate) is shown in Fig. 4 (top). By fitting an 8th order transfer function to the experimentally (impact hammer) measured unidirectional frequency response in [1], four complex pole pairs at $p_1 = 0.5775 \pm 0.8015i$, $p_2 = 0.7609 \pm 0.6347i$, $p_3 = 0.9596 \pm 0.2166i$ and $p_4 = 0.9924 \pm 0.0780i$ of the corresponding discrete-time model are obtained.

Systems based on the presented nominal model were subsequently used to assess the performance of different Kalman smoothers (RTS, MBF and BIFM), as well as of their respective square-root implementations (SR-MBF and SR-BIFM). In the numerical simulations the eigenvalues of the nominal system were randomly perturbed, however enforcing the magnitude of eigenvalues to be less than one, to guarantee stability. Thereby, a wide range of condition numbers for the steady-state covariance matrix was obtained (cf. Fig. 4).

In the (base-10) logarithmic domain, condition numbers were assigned to bins of size 0.5. For each bin, we ran 2000 simulations and successive state estimations at an average SNR of 24 dB and evaluated the resulting estimation failure rates. If the relative mean squared error of the state estimate exceeded 10% for any state, the estimation was considered as failed. Note that for computing the condition number of the steady-state covariance matrix, we used MATLAB's *dare*-function [22]. Systems for which the *dare*-function failed to find the steady-state solution, were not suited for comparison and were therefore discarded. Such systems were extremely rare and had a frequency of 61 out of one 10^6 simulated state space models. Remarkably however, despite the *dare*-function failing to obtain the steady-state covariance matrix (which precludes us from computing its condition number) the relative MSEs of the square-root MBF and square-root BIFM were small even for these kind of systems.

Figure 4 (center) shows a comparison in estimation failure rate between the RTS, MBF, square-root MBF (SR-MBF) and MBF after the state reparametrization (TS-MBF) suggested in subsection II-C, as a function of the condition number of the steady-state covariance. Figure 4 (bottom) shows the same comparison for the BIFM smoother. The superior performance of the regular BIFM smoother compared to the RTS and MBF in this scenario can partly be explained by the state space model being subject only to scalar input noise and absence of state noise. Additionally, in order to compare the BIFM with the MBF and RTS smoother, the failure rate is plotted against the condition number of the steady-state *forward covariance matrix*, which is correlated but not equal to the condition number of the backward precision matrix. The filtering step of the BIFM and SR-BFIM smoother however, are based on the information filter parametrizations.

The good performance of the Kalman smoothers based on the suggested state transformations, can be explained on the one hand by the reduction of the condition number of the state covariance matrix and on the other hand by the almost diagonal structure of the covariance matrix (cf. II-C). Furthermore, as shown in [23] state space models tend to have inferior numerical properties when they are in companion form, i.e. in controllable or observable canonical form. Therefore, a further contribution to numerical stability comes from the change of the state space model from a structured canonical form into a more balanced one. A major limitation is that the proposed transformation, unlike square-root message passing (cf. section III), works only for time-invariant systems and is based on (accurate) knowledge of the noise and system model. The excellent numerical performance and moderate increase in computational requirements, suggests therefore that square-root Kalman smoothers are the method of choice in numerically challenging scenarios.

IV. CONCLUSION

In practical applications which feature large dynamic ranges of the parameters, large state dimensions and high-precision

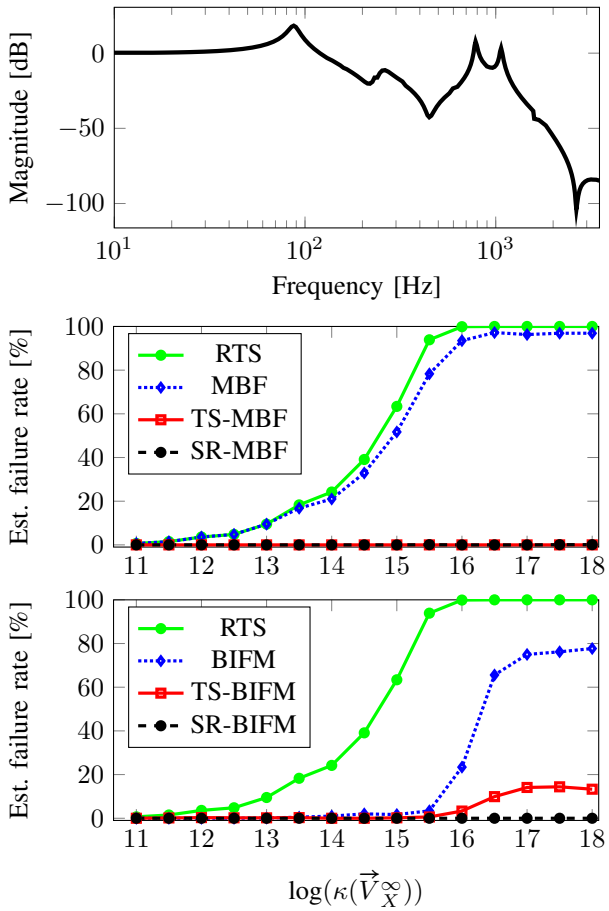


Fig. 4. Top: Nominal frequency response of the 4-mass resonator model fitted to experimental data from [1]. Center: Estimation failure rate for RTS, MBF, MBF with state reparametrization (TS-MBF) and square-root MBF (SR-MBF) plotted against the base-10 logarithm of the condition number. Bottom: Estimation failure rate for RTS, BIFM, BFIM with state reparametrization (TS-BIFM) and square-root BIFM (SR-BIFM).

measurements, usage of Kalman filters and smoothers is limited due to numerical stability issues. Instead of propagating the full covariance or precision matrix, propagating the Cholesky factors of these matrices, reduces the dynamic range and lowers the condition number in a principled way.

We presented a comprehensive list of tabulated square-root Gaussian message update rules (cf. Tables I-VI), from which a variety of both known and new square-root Kalman filters and smoothers can readily be composed. In particular, we presented a square-root version of the recently proposed BIFM smoother [5], which has a favorable performance in single-input, multiple-output (SIMO) systems, such as sensor networks. We furthermore suggested some additional improvements in the implementation of both the MBF and BIFM smoother and compared computational efficiency of these optimized versions.

Appendix: Derivation of Square-Root BIFM

Tables I-VI are the square-root version of the Gaussian message passing tables presented in [5], [8]. We will derive the updates rules of the square-root BIFM smoother (and therefore also for the square-root information filter) by identifying the Cholesky factor update rules with the (known) BIFM update rules for means, covariances and precisions from [5].

1) Equality Node (Table I): To obtain the Cholesky factor $\overleftarrow{S}_X$ and $\overleftarrow{\zeta}_X$ of (the time-reversed version of) (I.1) and (I.3) we compute the QR-decomposition of the following matrix, containing the Cholesky factors $\overleftarrow{S}_Y$, $\overleftarrow{S}_Z$, as well as $\overleftarrow{\zeta}_Y = \overleftarrow{S}_Y\overleftarrow{m}_Y$ and $\overleftarrow{\zeta}_Z = \overleftarrow{S}_Z\overleftarrow{m}_Z$:

$$Q\begin{pmatrix} \overleftarrow{S}_X & \overleftarrow{S}_X\overleftarrow{m}_X \\ 0 & * \end{pmatrix} = \mathrm{qr}\begin{pmatrix} \overleftarrow{S}_Z & \overleftarrow{S}_Z\overleftarrow{m}_Z \\ \overleftarrow{S}_Y & \overleftarrow{S}_Y\overleftarrow{m}_Y \end{pmatrix}. \quad (12)$$

We then left-multiply with the transpose of the matrix on the right-hand side of (12):

$$\begin{pmatrix} \overleftarrow{S}_X^{\mathsf{T}} & 0 \\ \overleftarrow{m}_X^{\mathsf{T}}\overleftarrow{S}_X^{\mathsf{T}} & * \end{pmatrix} Q^{\mathsf{T}}Q \begin{pmatrix} \overleftarrow{S}_X & \overleftarrow{S}_X\overleftarrow{m}_X \\ 0 & * \end{pmatrix} = \begin{pmatrix} \overleftarrow{S}_Z^{\mathsf{T}} & \overleftarrow{S}_Y^{\mathsf{T}} \\ \overleftarrow{m}_Z^{\mathsf{T}}\overleftarrow{S}_Z^{\mathsf{T}} & \overleftarrow{m}_Y^{\mathsf{T}}\overleftarrow{S}_Y^{\mathsf{T}} \end{pmatrix}\begin{pmatrix} \overleftarrow{S}_Z & \overleftarrow{S}_Z\overleftarrow{m}_Z \\ \overleftarrow{S}_Y & \overleftarrow{S}_Y\overleftarrow{m}_Y \end{pmatrix}. \quad (13)$$

Thereby we get:

$$\begin{pmatrix} \overleftarrow{S}_X^{\mathsf{T}}\overleftarrow{S}_X & \overleftarrow{S}_X^{\mathsf{T}}\overleftarrow{S}_X\overleftarrow{m}_X \\ \overleftarrow{m}_X^{\mathsf{T}}\overleftarrow{S}_X^{\mathsf{T}}\overleftarrow{S}_X & * \end{pmatrix} = \begin{pmatrix} \overleftarrow{S}_Z^{\mathsf{T}}\overleftarrow{S}_Z + \overleftarrow{S}_Y^{\mathsf{T}}\overleftarrow{S}_Y & \overleftarrow{S}_Z^{\mathsf{T}}\overleftarrow{S}_Z\overleftarrow{m}_Z + \overleftarrow{S}_Y^{\mathsf{T}}\overleftarrow{S}_Y\overleftarrow{m}_Y \\ \overleftarrow{m}_Z^{\mathsf{T}}\overleftarrow{S}_Z^{\mathsf{T}}\overleftarrow{S}_Z + \overleftarrow{m}_Y^{\mathsf{T}}\overleftarrow{S}_Y^{\mathsf{T}}\overleftarrow{S}_Y & * \end{pmatrix} \quad (14)$$

Noting $\overleftarrow{S}_X^{\mathsf{T}}\overleftarrow{S}_X = \overleftarrow{W}_X$, we identify entry (row = 1, col = 1) with $\overleftarrow{W}_X = \overleftarrow{W}_Y + \overleftarrow{W}_Z$ and entry (row = 1, col = 2) with $\overleftarrow{\xi}_X = \overleftarrow{\xi}_Y + \overleftarrow{\xi}_Z$. The invariance of the marginal covariance $V_X = V_Y = V_Z$ at the equality-node [8] carries over to its Cholesky factors from which (I.4) and (I.5) follow trivially.

2) Matrix Multiplier Node (Table III): To prove (III.3) we proceed as follows:

$$\overleftarrow{\xi}_X = \overleftarrow{W}_X\overleftarrow{m}_X = A^{\mathsf{T}}\overleftarrow{W}_Y\overleftarrow{m}_Y \quad (15)$$

$$\overleftarrow{S}_X^{\mathsf{T}}\overleftarrow{S}_X\overleftarrow{m}_X = A^{\mathsf{T}}\overleftarrow{S}_Y^{\mathsf{T}}(\overleftarrow{S}_Y\overleftarrow{m}_Y) = (\overleftarrow{S}_Y A)^{\mathsf{T}}(\overleftarrow{S}_Y\overleftarrow{m}_Y) \quad (16)$$

$$\overleftarrow{\zeta}_X \triangleq \overleftarrow{S}_X\overleftarrow{m}_X = \overleftarrow{S}_Y\overleftarrow{m}_Y \triangleq \overleftarrow{\zeta}_Y, \quad (17)$$

where (16) to (17) follows from (20). For the Cholesky factor of the precision (III.5), we have:

$$\overleftarrow{W}_X = A^{\mathsf{T}}\overleftarrow{W}_Y A \quad (18)$$

$$\overleftarrow{S}_X^{\mathsf{T}}\overleftarrow{S}_X = A^{\mathsf{T}}\overleftarrow{S}_Y^{\mathsf{T}}\overleftarrow{S}_Y A = (\overleftarrow{S}_Y A)^{\mathsf{T}}(\overleftarrow{S}_Y A) \quad (19)$$

$$\overleftarrow{S}_X = \overleftarrow{S}_Y A. \quad (20)$$

For the Cholesky factor of the marginal covariance (III.7) we have:

$$V_Y = AV_X A^{\mathsf{T}} \quad (21)$$

$$N_Y^{\mathsf{T}}N_Y = AN_X^{\mathsf{T}}N_X A^{\mathsf{T}} = (N_X A^{\mathsf{T}})^{\mathsf{T}}(N_X A^{\mathsf{T}}) \quad (22)$$

$$N_Y = N_X A^{\mathsf{T}}. \quad (23)$$

3) Input Block (Table V): To obtain the Cholesky factor of $\overleftarrow{S}_X$ at the input block we start with:

$$Q \begin{pmatrix} \tilde{H} & \tilde{R} \\ 0 & \overleftarrow{S}_X \end{pmatrix} = \begin{pmatrix} \overrightarrow{S}_Y & 0 \\ \overleftarrow{S}_Z A & \overleftarrow{S}_Z \end{pmatrix} \tag{24}$$

We get:

$$\begin{pmatrix} \tilde{H}^\mathsf{T} & 0 \\ \tilde{R}^\mathsf{T} & \overleftarrow{S}_X \end{pmatrix} \begin{pmatrix} \tilde{H} & \tilde{R} \\ 0 & \overleftarrow{S}_X \end{pmatrix} = \begin{pmatrix} \overrightarrow{S}_Y^\mathsf{T} & A^\mathsf{T}\overleftarrow{S}_Z^\mathsf{T} \\ 0 & \overleftarrow{S}_Z^\mathsf{T} \end{pmatrix} \begin{pmatrix} \overrightarrow{S}_Y & 0 \\ \overleftarrow{S}_Z A & \overleftarrow{S}_Z \end{pmatrix} \tag{25}$$

$$\begin{pmatrix} \tilde{H}^\mathsf{T}\tilde{H} & \tilde{H}^\mathsf{T}\tilde{R} \\ \tilde{R}^\mathsf{T}\tilde{H} & \tilde{R}^\mathsf{T}\tilde{R}+\overleftarrow{S}_X^\mathsf{T}\overleftarrow{S}_X \end{pmatrix} = \begin{pmatrix} \overrightarrow{S}_Y^\mathsf{T}\overrightarrow{S}_Y + A^\mathsf{T}\overleftarrow{S}_Z^\mathsf{T}\overleftarrow{S}_Z A & A^\mathsf{T}\overleftarrow{S}_Z^\mathsf{T}\overleftarrow{S}_Z \\ \overleftarrow{S}_Z^\mathsf{T}\overleftarrow{S}_Z A & \overleftarrow{S}_Z^\mathsf{T}\overleftarrow{S}_Z \end{pmatrix} \tag{26}$$

We can identify $\tilde{H}$ as the Cholesky factor of $\overrightarrow{W}_Y + A^\mathsf{T}\overleftarrow{W}_Z A$, i.e. the Cholesky factor of H^{-1} from [8]. With $\tilde{R} = \tilde{H}^{-\mathsf{T}}A^\mathsf{T}\overleftarrow{W}_Z$ and finally we identify entry (row = 2, col = 2) with $\overleftarrow{W}_X = \overleftarrow{W}_Z - \tilde{R}^\mathsf{T}\tilde{R} = \overleftarrow{W}_Z - \overleftarrow{W}_Z A(\tilde{H}^\mathsf{T}\tilde{H})^{-1}A^\mathsf{T}\overleftarrow{W}_Z$, which is the precision update formula (cf. [8]).

To obtain $\overleftarrow{\zeta}_X$ at the input block we start with the formula of the mean update:

$$\overleftarrow{m}_X = \overleftarrow{m}_Z - A\overrightarrow{m}_Y \tag{27}$$

$$\overleftarrow{S}_X\overleftarrow{m}_X = \overleftarrow{S}_X\overleftarrow{m}_Z - \overleftarrow{S}_X A\overrightarrow{m}_Y \tag{28}$$

$$\overleftarrow{S}_X\overleftarrow{m}_X = Q_2^\mathsf{T}\begin{pmatrix} 0 \\ \overleftarrow{S}_Z \end{pmatrix}\overleftarrow{m}_Z - Q_2^\mathsf{T}\begin{pmatrix} 0 \\ \overleftarrow{S}_Z \end{pmatrix} A\overrightarrow{m}_Y \tag{29}$$

$$\overleftarrow{\zeta}_X \triangleq \overleftarrow{S}_X\overleftarrow{m}_X = Q_2^\mathsf{T}\begin{pmatrix} 0 \\ \overleftarrow{\zeta}_Z - \overleftarrow{S}_Z A\overrightarrow{m}_Y \end{pmatrix}. \tag{30}$$

From [5] we have:

$$\tilde{F} \triangleq I - \overleftarrow{W}_Z AHA^\mathsf{T} = I - \overleftarrow{S}_Z^\mathsf{T}\overleftarrow{S}_Z A(\tilde{H}^\mathsf{T}\tilde{H})^{-1}A^\mathsf{T}. \tag{31}$$

To get the Cholesky factor N_Z of the marginal covariance V_Z we proceed as follows:

$$Q\begin{pmatrix} N_X \\ 0 \end{pmatrix} = \begin{pmatrix} N_Z\tilde{F} \\ \tilde{H}^{-\mathsf{T}}A^\mathsf{T} \end{pmatrix}. \tag{32}$$

Now following the lines of argumentation used for the square-root MBF in [18], we left multiply with the transpose of that matrix and compare the result with the standard BIFM formulas:

$$\left(N_X^\mathsf{T}, 0\right)\begin{pmatrix} N_X \\ 0 \end{pmatrix} = \left(\tilde{F}^\mathsf{T}N_Z^\mathsf{T}, A\tilde{H}^{-1}\right)\begin{pmatrix} N_Z\tilde{F} \\ \tilde{H}^{-\mathsf{T}}A^\mathsf{T} \end{pmatrix} \tag{33}$$

$$N_X^\mathsf{T}N_X = \tilde{F}^\mathsf{T}N_Z^\mathsf{T}N_Z\tilde{F} + A(\tilde{H}^\mathsf{T}\tilde{H})^{-1}A^\mathsf{T}. \tag{34}$$

Identifying $V_X = N_X^\mathsf{T}N_X$ and $H = (\tilde{H}^\mathsf{T}\tilde{H})^{-1}$ yields:

$$V_X = \tilde{F}^\mathsf{T}V_Z\tilde{F} + AHA^\mathsf{T}, \tag{35}$$

which concludes the proof.

Finally, to obtain the update for the mean, we start with the standard BIFM update:

$$m_Z = \tilde{F}^\mathsf{T}m_X + AH\left(A^\mathsf{T}\overleftarrow{\xi}_Z + \overrightarrow{\xi}_Y\right) \tag{36}$$

from [5] and replace the required quantities with the computed Cholesky factors:

$$m_Z = \tilde{F}^\mathsf{T}m_X + A(\tilde{H}^\mathsf{T}\tilde{H})^{-1}\left(A^\mathsf{T}\overleftarrow{S}_Z^\mathsf{T}\overleftarrow{\zeta}_Z + \overrightarrow{S}_Y^\mathsf{T}\overrightarrow{\zeta}_Y\right). \tag{37}$$

REFERENCES

[1] L. Bruderer, *Input Estimation and Dynamical System Identification: New Algorithms and Results*, Ph.D. thesis, Eidgenössische Technische Hochschule ETH Zürich, Nr. 22575, 2015.

[2] F. Wadehn, L. Bruderer, D. Waltisberg, T. Keresztfalvi, and H.-A. Loeliger, "Sparse-input detection algorithm with applications in electrocardiography and ballistocardiography," in *International Conf. on Bio-inspired Systems and Signal Processing*, 2015.

[3] H. E. Rauch, C.T. Striebel, and F. Tung, "Maximum likelihood estimates of linear dynamic systems," *AIAA Journal*, vol. 3, no. 8, 1965.

[4] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear Estimation*, vol. 1, Prentice Hall Upper Saddle River, NJ, 2000.

[5] H.-A. Loeliger, L. Bruderer, H. Malmberg, F. Wadehn, and N. Zalmai, "On sparsity by NUV-EM, Gaussian message passing, and Kalman smoothing," *arXiv preprint arXiv:1602.02673*, 2016.

[6] P. G. Kaminski, A. E. Bryson Jr, and S. F. Schmidt, "Discrete square root filtering: A survey of current techniques," *IEEE Transactions on Automatic Control*, vol. 16, no. 6, pp. 727–736, 1971.

[7] L. Bruderer, H. Malmberg, and H.-A. Loeliger, "Deconvolution of weakly-sparse signals and dynamical-system identification by gaussian message passing," in *International Symposium on Information Theory (ISIT)*. IEEE, 2015, pp. 326–330.

[8] H.-A. Loeliger, J. Dauwels, J. Hu, S. Korl, L. Ping, and F. R. Kschischang, "The factor graph approach to model-based signal processing," *Proceedings of the IEEE*, vol. 95, no. 6, pp. 1295–1322, 2007.

[9] L. Bolliger, H.-A. Loeliger, and C. Vogel, "LMMSE estimation and interpolation of continuous-time signals from discrete-time samples using factor graphs," *arXiv preprint arXiv:1301.4793*, 2013.

[10] F. Wadehn, L. Bruderer, J. Dauwels, H. Yu, V. Sahdeva, and H.-A. Loeliger, "Outlier-insensitive Kalman smoothing and marginal message passing," in *Proc. 2016 European Signal Proc. Conf. (EUSIPCO)*, Budapest, Hungary, Aug. 29-Sept. 2, 2016.

[11] G. H. Golub and C. F. Van Loan, *Matrix Computations*, vol. 3, JHU Press, 2012.

[12] M. Verhaegen and P. Van Dooren, "Numerical aspects of different Kalman filter implementations," *IEEE Transactions on Automatic Control*, vol. 31, no. 10, pp. 907–917, 1986.

[13] K. Underwood, "FPGAs vs. CPUs: trends in peak floating-point performance," in *Proceedings of the 12th ACM/SIGDA International Symposium on Field Programmable Gate Arrays*. ACM, 2004, pp. 171–180.

[14] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation: theory algorithms and software*, John Wiley & Sons, 2004.

[15] S. C. Chapra and R. P. Canale, *Numerical methods for engineers*, vol. 2, McGraw-Hill New York, 2012.

[16] S. Liu, E. Masazade, M. Fardad, and P. K. Varshney, "Sparsity-aware field estimation via ordinary kriging," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 3948–3952.

[17] T. Gudmundsson, C. Kenney, and A. J. Laub, "Scaling of the discrete-time algebraic Riccati equation to enhance stability of the Schur solution method," *IEEE Transactions on Automatic Control*, vol. 37, no. 4, pp. 513–518, 1992.

[18] R. G. Gibbs, "Square root Modified Bryson–Frazier smoother," *IEEE Transactions on Automatic Control*, vol. 56, no. 2, pp. 452–456, 2011.

[19] B. D. Anderson and J. B. Moore, *Optimal filtering*, Courier Corporation, 2012.

[20] F. Girardin, D. Remond, and J.-F. Rigal, "High frequency correction of dynamometer for cutting force observation in milling," *Journal of Manufacturing Science and Engineering*, vol. 132, no. 3, 2010.

[21] S. S. Park and Y. Altintas, "Dynamic compensation of spindle integrated force sensors with Kalman filter," *Journal of Dynamic Systems, Measurement, and Control*, vol. 126, no. 3, pp. 443–452, 2004.

[22] MATLAB, *8.5.0.197613 (R2015a)*, The MathWorks Inc., Natick, Massachusetts, 2015.

[23] Michel Verhaegen and Vincent Verdult, *Filtering and system identification: a least squares approach*, Cambridge University Press, 2007.