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—Naive implementations of Kalman filters and smoothers often suffer from numerical problems. In this paper, we consider two Kalman smoothers that were proposed recently: (i) the adaptation of the MBF (Modified-Bryson Frazier) for input estimation and (ii) the BIFM (backward information filter, forward marginal) smoother. Even naive implementations of these smoothers are numerically rather robust because these smoothers require no matrix inversion. Nonetheless, additional measures are sometimes required. We present square-root versions for both these smoothers as well as state reparametrizations for improved numerical stability. 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 with 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: First, on the Modified-Bryson-Frazier (MBF) [4] and its extension to input estimation [5] and second, 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 with 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 [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 \( y = (y_1, \ldots, y_N) \) with \( y_k \in \mathbb{R}^d \), will be modeled as the output of a linear state space model (SSM):

\[
X_k = AX_{k-1} + BU_{k-1} + W_{k-1}
Y_k = CX_k + Z_k,
\]

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

\[
p(y_{1:N}, x_{0:N}|u_{0:N-1}) = p(x_0) \prod_{k=1}^N p(y_k|x_k)p(x_k|x_{k-1}, u_{k-1})
\]

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/smoothers 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 ranging from state estimation, to output smoothing and input estimation as shown in Section III-B.
Fig. 1. Factor graph segment of the state space model. The whole factor graph consists of many such sections, one for each time step.

B. Message Passing Notation

The (forward) filtering distribution \( p(x_k | y_1, \ldots, y_k) \), e.g., computed via the covariance Kalman filter, is parameterized by the forward mean \( \hat{m}_{X_k} \) and the forward covariance matrix \( \hat{V}_{X_k} \). The (backward) filtering distribution \( p(x_k | y_1, \ldots, y_N) \), e.g., computed via the backward information filter, is parameterized by the backward transformed mean \( \Xi_{X_k} = \hat{W}_{X_k} \hat{m}_{X_k} \) and backward precision \( \hat{W}_{X_k} = \hat{V}_{X_k}^{-1} \). 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 \( p(x_k | y_1, \ldots, y_N) \) is parameterized via the mean \( m_{X_k} \) and covariance \( V_{X_k} \) (BIFM smoother) or alternatively by the dual mean \( \hat{\xi}_{X_k} \) and dual precision \( \hat{W}_{X_k} \) (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 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 \( (Av)v^T = A(vv^T) \) are mathematically equivalent, however the left-hand side involves one matrix-vector multiplication \( O(n^2) \) and one outer product \( O(n^3) \), 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 \( \hat{W}_{X_k} \), through the A-node and observation block (equality- and C-node) in Fig. 1 yields:

\[
\hat{W}_{X_{k-1}} = A^T F^k_k \hat{W}_{X_k} F_k A + A^T C^T G_k C A,
\]

where \( F_k \) and \( G_k \) (cf. Table IV) are summary quantities obtained in the Kalman filtering step. By introducing \( F^k_k = F_k A \), 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:

\[
F^a_k = F_k A = (I - \hat{V}_{X_k}^{\top} C^T G_k C) A.
\]

For the BIFM smoother (being dual to the MBF) introducing the auxiliary variable \( \hat{F}^a_k = A^T \hat{F}_k \) 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

Naive 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
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:

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

where $\rho$ 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 $\sigma_Z$ 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-manoeuvring target is modeled by a continuous-time white noise acceleration model, giving rise to the discrete-time system:

$$A = \begin{bmatrix} 0.8 & 0 & \Delta/10 & 0; 0, 0.8, 0, \Delta/10; 0, 0, 0.8, 0, 0, 0, 0, 0.8 \end{bmatrix}, \quad B = 0.$$  

The state noise covariance was chosen as in [16]:

$$\Sigma_{W} = \tau(\Delta^3/3, 0, \Delta^2/2, 0, 0, \Delta^3/3, 0, \Delta^2/2, 2, \Delta, 0, 0, 0, 0, 0, 0, \Delta).$$

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 $\Delta$, denotes the sampling time and is set equal to 0.1 and $\tau$ 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 $C$ matrix are independently drawn from a multivariate standard Normal distribution as in [16].

For each noise scaling factor level $\sigma_Z$ 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 $\langle ||x - \bar{x}||/||x|| \rangle$ 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 $\sigma_Z$. 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 $(V + V^T)/2$ 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].

In Kalman filtering and smoothing, 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

$$\tilde{V}_X = QAQ^T,$$

with $Q$ being the eigenbasis and $A$ a diagonal matrix containing the eigenvalues. Thereby we obtain the transformation matrix $T = \Lambda^{-\frac{1}{2}}Q^T$ (with $\bar{x} = TX$), such that the covariance matrix of the transformed state $\tilde{V}_{\tilde{X}_k} = TV_{X_k}T^T \approx \Lambda^2$ 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 $\tilde{W}_X$ 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 semi-definite matrices $A$ and $B$ there always exists an invertible matrix (cf. page 500 in [11]), such that $T^TAT$ is diagonal and $T^TB$ 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 $\tilde{V}_X$ and the steady-state (backwards) precision $\tilde{W}_X^\ast$ 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 $\tilde{V}_{\tilde{X}_k} = TV_{X_k}T^T$ and backward precision $\tilde{W}_{\tilde{X}_k} = T^{-T}\tilde{W}_{X_k}T^{-1}$ 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 $V_X = N^T_XN_X$ and $W_X = S^T_XS_X$, 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 $N_X^\ast$ and $S_X^\ast$. In the final step, these factors are combined at each node to obtain (symmetric and positive semidefinite) covariance $V_X^\ast$ and precision matrices $W_X^\ast$ [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 $\xi_X = \bar{W}_Xm_X^\ast$ as shown in [14] and the other which uses a factored mean $\xi_X = \bar{Z}_Xm_X^\ast$. 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: $C^TCA + BTB$ (e.g. covariance update at the plus-node or precision update at equality-node) and $C^TCA - BTB$ (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 update equations of the form $C^TCA = A^T + BTB$, we proceed as follows:

$$C^TCA = A^T + BTB$$
$$\begin{bmatrix} C & 0 \end{bmatrix} = \begin{bmatrix} A^T & BT \end{bmatrix}$$

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

$$Q \begin{bmatrix} C & 0 \end{bmatrix} = qr \begin{bmatrix} A & B \end{bmatrix}.$$
Constraint $X = Y = Z$, expressed by $\delta(z - x)\delta(y - x)$. Forward precision $\hat{W} = \hat{S}^T \hat{S}$ and covariance $V = N^T N$.

$$\zeta_Z = Q_1^T \left( \begin{array}{c} \zeta_X \\ \zeta_Y \end{array} \right)$$ (I.1)

$$\zeta_Z = \zeta_X + \zeta_Y$$ (I.2)

$$(Q_1, Q_2) \left( \begin{array}{c} \bar{S}_Z \\ 0 \end{array} \right) = \text{qr} \left( \begin{array}{c} \bar{S}_X \\ \bar{S}_Y \end{array} \right)$$ (I.3)

$$m_X = m_Y = m_Z$$ (I.4)

$$N_X = N_Y = N_Z$$ (I.5)

$$\xi_X = \xi_Y + \xi_Z$$ (I.6)

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

The 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).

The Covariance Square-Root Filter [19]: is based on the forward recursion with the forward mean $\bar{m}_{X_k}$ and the Cholesky factor $\bar{N}_{X_k}$ of the forward covariance $\bar{V}_{X_k} = \bar{N}_{X_k}^T \bar{N}_{X_k}$. The mean and Cholesky factor are propagated through the A-matrix using (III.1) and (III.2) respectively. The input and state noise are accounted for with (V.1) and (V.3). The update step

$$\zeta_X = \zeta_Y = \zeta_Z$$ (I.6)

$$m_X = m_Y = m_Z$$ (I.4)

$$N_X = N_Y = N_Z$$ (I.5)

$$\xi_X = \xi_Y + \xi_Z$$ (I.6)

The Covariance Square-Root Filter [19]: is based on the forward recursion with the forward transformed mean $\bar{m}_{X_k}$ and the Cholesky factor $\bar{S}_{X_k}$ of the forward precision $\bar{W}_{X_k} = \bar{S}_{X_k}^T \bar{S}_{X_k}$. The transformed mean and Cholesky factor are propagated through the A-matrix using (III.1) and (III.2) respectively. The input and state noise are accounted for with (V.1) and (V.3). The update step

$$\zeta_X = \zeta_Y = \zeta_Z$$ (I.6)

$$m_X = m_Y = m_Z$$ (I.4)

$$N_X = N_Y = N_Z$$ (I.5)

$$\xi_X = \xi_Y + \xi_Z$$ (I.6)

The Covariance Square-Root Filter [19]: is based on the forward recursion with the forward transformed mean $\bar{m}_{X_k}$ and the Cholesky factor $\bar{S}_{X_k}$ of the forward precision $\bar{W}_{X_k} = \bar{S}_{X_k}^T \bar{S}_{X_k}$. The transformed mean and Cholesky factor are propagated through the A-matrix using (III.1) and (III.2) 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.

The 2-Filter Smoother: is based on combining the forward mean $\bar{m}_{X_k}$ and the Cholesky factor $\bar{N}_{X_k}$ with the backward transformed mean $\bar{\zeta}_{X_k}$ and the Cholesky factor $\bar{A}$.
with $\widetilde{S}_{X_k}$ 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 $V_X = N_X^T N_X$ as follows:

$$m_X = \widetilde{m}_X + M^{-T} \widetilde{L}^T (\widetilde{\zeta}_X - \widetilde{S}_X \widetilde{m}_X) \quad (10)$$

$$Q \left( \begin{array}{c} \tilde{L} \\ \tilde{M} \\ 0 \\ N_X \end{array} \right) = qr \left( \begin{array}{cccc} I & 0 \\ \tilde{N}_X & \tilde{S}_X \\ 0 & \tilde{N}_X \end{array} \right). \quad (11)$$

**d)** The Square-Root MBF [18]: is based on the forward recursion using the Square-Root Covariance Filter followed by propagating the dual mean $\hat{\xi}_{X_k}$ and the Cholesky factor of the dual precision $\hat{S}_{X_k}$ 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 $\bar{m}_{X_k}$ and the Cholesky factor of the variance $\bar{N}_{X_k}$ with $V_{X_k} = N_{X_k}^T N_{X_k}$ 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 [1], [9]. Here we propose numerically stable square-root versions of this input estimator.

The square-root MBF input estimator computes the dual mean $\hat{\xi}_{U_k}$ and Cholesky factor $\hat{S}_{U_k}$ 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 $\hat{\xi}_{U_k}$ and Cholesky factor $\hat{S}_{U_k}$) with the Gaussian prior on the input (parametrized by $\overline{m}_{U_k}$ and $V_{U_k}$).
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 unidirectional frequency response in [1], four complex pole 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\textsuperscript{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
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 \( \mathbf{\widetilde{S}}_X \) and \( \mathbf{\widetilde{Y}}_X \) of (the time-reversed version of) (I.1) and (I.3) we compute the QR-decomposition of the following matrix, containing the Cholesky factors \( \mathbf{\widetilde{S}}_Y, \mathbf{\widetilde{S}}_Z \), as well as \( \mathbf{\widetilde{z}}_Y = \mathbf{\widetilde{S}}_Y \mathbf{\widetilde{m}}_Y \) and \( \mathbf{\widetilde{z}}_Z = \mathbf{\widetilde{S}}_Z \mathbf{\widetilde{m}}_Z \):

\[
Q \begin{pmatrix} \mathbf{\widetilde{S}}_X & \mathbf{\widetilde{m}}_X \\ 0 & \mathbf{\widetilde{m}}_X \end{pmatrix} = qr \begin{pmatrix} \mathbf{\widetilde{S}}_Z & \mathbf{\widetilde{m}}_Z \\ \mathbf{\widetilde{S}}_Y & \mathbf{\widetilde{m}}_Y \end{pmatrix}.
\] (12)

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

\[
\begin{pmatrix} \mathbf{\widetilde{S}}_X^T & \mathbf{\widetilde{m}}_X^T \\ 0 & 0 \end{pmatrix} Q^T Q \begin{pmatrix} \mathbf{\widetilde{S}}_X & \mathbf{\widetilde{m}}_X \end{pmatrix} = \begin{pmatrix} \mathbf{\widetilde{S}}_Z & \mathbf{\widetilde{m}}_Z \\ \mathbf{\widetilde{S}}_Y & \mathbf{\widetilde{m}}_Y \end{pmatrix}.
\] (13)

Thereby we get:

\[
\begin{pmatrix} \mathbf{\widetilde{S}}_X^T \mathbf{\widetilde{S}}_X & \mathbf{\widetilde{m}}_X^T \mathbf{\widetilde{S}}_X \\ \mathbf{\widetilde{m}}_X^T \mathbf{\widetilde{S}}_X & \mathbf{\widetilde{m}}_X \end{pmatrix} = \begin{pmatrix} \mathbf{\widetilde{S}}_Z^T \mathbf{\widetilde{S}}_Z + \mathbf{\widetilde{S}}_Y^T \mathbf{\widetilde{S}}_Y \\ \mathbf{\widetilde{S}}_Z \mathbf{\widetilde{m}}_Z + \mathbf{\widetilde{S}}_Y \mathbf{\widetilde{m}}_Y \end{pmatrix}.
\] (14)

Noting \( \mathbf{\widetilde{S}}_X^T \mathbf{\widetilde{S}}_X = \mathbf{\widetilde{W}}_X \), we identify entry (row = 1, col = 1) with \( \mathbf{\widetilde{W}}_X = \mathbf{\widetilde{W}}_Y + \mathbf{\widetilde{W}}_Z \) and entry (row = 1, col = 2) with \( \xi_X = \xi_Y + \xi_Z \). 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 (14) and (15) follow trivially.

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

\[
\bar{\xi}_X = \mathbf{\widetilde{W}}_X \mathbf{\widetilde{m}}_X = A^T \mathbf{\widetilde{W}}_Y \mathbf{\widetilde{m}}_Y \quad \text{(15)}
\]

\[
\bar{\mathbf{S}}_X \mathbf{\widetilde{S}}_X \mathbf{\widetilde{m}}_X = A^T \mathbf{\widetilde{S}}_Y (\mathbf{\widetilde{S}}_Y \mathbf{\widetilde{m}}_Y) = (\mathbf{\widetilde{S}}_Y A)^T (\mathbf{\widetilde{S}}_Y A) \quad \text{(16)}
\]

\[
\bar{\xi}_X = \mathbf{\widetilde{S}}_Y \mathbf{\widetilde{m}}_X = \mathbf{\widetilde{S}}_Y \mathbf{\widetilde{m}}_Y = \chi_Y \quad \text{(17)}
\]

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

\[
\mathbf{\widetilde{W}}_X = A^T \mathbf{\widetilde{W}}_Y A \quad \text{(18)}
\]

\[
\mathbf{\widetilde{S}}_X \mathbf{\widetilde{S}}_X = A^T \mathbf{\widetilde{S}}_Y A (\mathbf{\widetilde{S}}_Y A)^T = (\mathbf{\widetilde{S}}_Y A)^T (\mathbf{\widetilde{S}}_Y A) \quad \text{(19)}
\]

\[
\mathbf{\widetilde{S}}_X = \mathbf{\widetilde{S}}_Y A. \quad \text{(20)}
\]

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

\[
V_Y = AV_X A^T \quad \text{(21)}
\]

\[
N_Y = AN_X A^T = (N_X A^T)^T (N_X A^T) \quad \text{(22)}
\]

\[
N_Y = N_X A^T. \quad \text{(23)}
\]
3) Input Block (Table V): To obtain the Cholesky factor of $\tilde{S}_X$ at the input block we start with:

$$Q \left( \tilde{H} \tilde{R} \begin{bmatrix} \tilde{S}_Y \\ \tilde{S}_Z \end{bmatrix} \right) = \begin{pmatrix} \tilde{S}_Y \\ \tilde{S}_Z A \\ 0 \\ \tilde{S}_Z \end{pmatrix}$$  \hfill (24)

We get:

$$\begin{pmatrix} \tilde{H}^T & \tilde{R} \end{pmatrix} \begin{pmatrix} \tilde{H} & \tilde{R} \end{pmatrix} = \begin{pmatrix} \tilde{S}_Y + A^T \tilde{S}_Z A & \tilde{S}_Y \\ \tilde{S}_Z A & \tilde{S}_Z \end{pmatrix}$$ \hfill (25)

$$\begin{pmatrix} \tilde{H}^T & \tilde{R} \end{pmatrix} \begin{pmatrix} \tilde{H}^T & \tilde{R} \end{pmatrix} = \begin{pmatrix} \tilde{S}_Y + A^T \tilde{S}_Z A & \tilde{S}_Y \\ \tilde{S}_Z A & \tilde{S}_Z \end{pmatrix}$$ \hfill (26)

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

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

$$\tilde{m}_X = m_Z - \tilde{A} \tilde{m}_Y$$ \hfill (27)

$$\tilde{S}_X \tilde{m}_X = \tilde{S}_X \tilde{m}_Y - \tilde{S}_X \tilde{A} \tilde{m}_Y$$ \hfill (28)

$$\tilde{S}_X \tilde{m}_X = Q_2 \begin{pmatrix} \tilde{m}_Z \\ \tilde{m}_Y \end{pmatrix} + Z \begin{pmatrix} \tilde{m}_Z \\ \tilde{m}_Y \end{pmatrix}$$ \hfill (29)

$$\tilde{S}_X \tilde{m}_X = Q_2 \begin{pmatrix} \tilde{m}_Z \\ \tilde{m}_Y \end{pmatrix} + Z \begin{pmatrix} \tilde{m}_Z \\ \tilde{m}_Y \end{pmatrix}$$ \hfill (30)

$$\tilde{\zeta}_X = \tilde{m}_X - \tilde{m}_Z A \tilde{m}_Y$$ \hfill (31)

From [5] we have:

$$\tilde{F} \equiv I - \tilde{W}_Z A H \tilde{A}^T = I - \tilde{S}_X A (H^T \tilde{H})^{-1} A^T.$$ \hfill (32)

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

$$Q \begin{pmatrix} N_X \\ 0 \end{pmatrix} = \begin{pmatrix} N_Z \tilde{F} \\ H^{-1} A^T \end{pmatrix}.$$ \hfill (33)

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:

$$N_X^T N_X = \tilde{F}^T \tilde{N}_Z N_X \tilde{F} + A (H^T \tilde{H})^{-1} A^T.$$ \hfill (34)

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

$$V_X = \tilde{F}^T V_Z \tilde{F} + A H A^T,$$ \hfill (36)

which concludes the proof.

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

$$m_Z = \tilde{F}^T m_X + A (H^T \tilde{H})^{-1} \left( A^T \tilde{S}_Y \tilde{F} + \tilde{S}_Y \tilde{F} \right).$$ \hfill (37)

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

$$m_Z = \tilde{F}^T m_X + A (H^T \tilde{H})^{-1} \left( A^T \tilde{S}_Y \tilde{F} + \tilde{S}_Y \tilde{F} \right).$$ \hfill (38)

References


[22] MATLAB, 8.5.0.179613 (R2015a), The MathWorks Inc., Natick, Massachusetts, 2015.