Factored-form Kalman-like Implementations under Maximum Correntropy Criterion

Maria V. Kulikova^{a,*}

^aCEMAT (Center for Computational and Stochastic Mathematics), Instituto Superior Técnico, Universidade de Lisboa, Av. Rovisco Pais 1, 1049-001 Lisboa, Portugal

Abstract

The maximum correntropy criterion (MCC) methodology is recognized to be a robust filtering strategy with respect to outliers and shown to outperform the classical Kalman filter (KF) for estimation accuracy in the presence of non-Gaussian noise. However, the numerical stability of the newly proposed MCC-KF estimators in finite precision arithmetic is seldom addressed. In this paper, a family of *factored-form* (square-root) algorithms is derived for the MCC-KF and its improved variant, respectively. The family traditionally consists of three factored-form implementations: (i) Cholesky factorization-based algorithms, (ii) modified Cholesky, i.e. UD-based methods, and (iii) the recently established SVD-based filtering. All these strategies are commonly recognized to enhance the numerical robustness of conventional filtering with respect to roundoff errors and, hence, they are the preferred implementations when solving applications with high reliability requirements. Previously, only Cholesky-based IMCC-KF algorithms have been designed. This paper enriches a factored-form family by introducing the UD- and SVD-based methods as well. A special attention is paid to *array* algorithms that are proved to be the most numerically stable and, additionally, suitable for parallel implementations. The theoretical properties are discussed and numerical comparison is presented for determining the most reliable implementations.

Keywords: Maximum correntropy filtering, square-root algorithms, Cholesky factorization, singular value decomposition.

1. Introduction and problem statement

Consider a linear discrete-time stochastic system

$$x_{k} = F_{k-1}x_{k-1} + G_{k-1}w_{k-1}, \qquad k \ge 1$$
(1)
$$y_{k} = H_{k}x_{k} + v_{k}$$
(2)

where $F_k \in \mathbb{R}^{n \times n}$, $G_k \in \mathbb{R}^{n \times q}$, $H_k \in \mathbb{R}^{m \times n}$. The vectors $x_k \in \mathbb{R}^n$ and $y_k \in \mathbb{R}^m$ are the unknown dynamic state and the available measurements, respectively. The random variables $\{x_0, w_k, v_k\}$ have the following properties:

$$\mathbf{E}\left\{\begin{bmatrix}x_{0}\\w_{k}\\v_{k}\end{bmatrix}\begin{bmatrix}x_{0}^{T}&w_{j}^{T}&v_{j}^{T}&1\end{bmatrix}\right\} = \begin{bmatrix}\Pi_{0} & 0 & 0 & \bar{x}_{0}\\0 & Q_{k}\delta_{kj} & 0 & 0\\0 & & R_{k}\delta_{kj} & 0\end{bmatrix}$$

where $Q_k \in \mathbb{R}^{q \times q}$, $R_k \in \mathbb{R}^{m \times m}$ and δ_{kj} denotes the Kronecker delta function.

The goal of any filtering method is to recover the unknown random sequence $\{x_k\}_1^N$ from the observed one $\{y_k\}_1^N$. The classical Kalman filter (KF) yields the minimum *linear* expected mean square error (MSE) estimate $\hat{x}_{k|k}$ of the state vector x_k , given measurements $\mathcal{Y}_1^k = \{y_1, \dots, y_k\}$, i.e.

$$\underset{\hat{x}_{k|k} \in span(\mathcal{Y}_1^k)}{\arg\min} \mathbf{E}\left\{ \|x_k - \hat{x}_{k|k}\|^2 \right\}.$$
(3)

If the examined state-space model is Gaussian, i.e. w_k , v_k and the initial state x_0 are jointly Gaussian, then estimator (3)

*Corresponding author. Email address: maria.kulikova@ist.utl.pt (Maria V. Kulikova)

Preprint submitted to Elsevier

coincides with the minimum expected MSE estimator [1].

$$\arg\min_{\hat{x}_{k|k}} \mathbf{E} \left\{ \|x_k - \hat{x}_{k|k}\|^2 \right\}.$$
(4)

In other words, although the classical KF is a linear estimator, in Gaussian settings it yields the minimum expected MSE estimate. In general (non-Gaussian case), the classical KF produces only sub-optimal solution for estimation problem (4). To deal with non-Gaussian uncertainties and outliers/impulsive noise in state-space model (1), (2), various "distributional robust" filtering/smoothing methods have been developed in engineering literature. For detecting outliers, the Huber-based and M-estimator-based KF algorithms suggest to construct weight matrices (or scalars) and utilize them for inflating the innovation or measurement noise covariances for reducing the estimation error [2, 3, 4]. The unknown input filtering (UIF) methodology suggests to model unknown external excitations as unknown inputs and, next, to derive the robust observer [5, 6]. Meanwhile, the most recent and comprehensive survey of existed Kalman-like smoothing methods developed for non-Gaussian state-space models can be found in [1]. In this paper, an alternative strategy called the maximum correntropy criterion (MCC) filtering is in the focus. It becomes an important topic for analysis in the past few years, both for linear [7, 8, 9, 10, 11, 12] and nonlinear systems [13, 14, 15].

The correntropy represents a similarity measure of two random variables. It can be used as an optimization cost in related estimation problem as discussed in [16, Chapter 5]: an estimator of unknown state $X \in \mathbb{R}$ can be defined as a function of observations $Y \in \mathbb{R}^m$, i.e. $\hat{X} = g(Y)$ where g is solved by maximizing the correntropy between X and \hat{X} that is [17]

$$g_{MCC} = \arg\max_{g \in G} V(X, \hat{X}) = \arg\max_{g \in G} \mathbf{E} \left\{ k_{\sigma} \left(X - g(Y) \right) \right\}$$
(5)

where *G* stands for the collection of all measurable functions of *Y*, $k_{\sigma}(\cdot)$ is a kernel function and $\sigma > 0$ is the kernel size (bandwidth). One of the most popular kernel function utilized in practice is the Gaussian kernel given as follows:

$$k_{\sigma}(X - \hat{X}) = \exp\left\{-(X - \hat{X})^2/(2\sigma^2)\right\}.$$
 (6)

It is not difficult to see that the MCC cost (5) with Gaussian kernel (6) reaches its maximum if and only if $X = \hat{X}$.

In [8, 11], the MCC estimation problem (5) with kernel (6) is combined with the minimum *linear* expected MSE estimation problem related to the classical KF in (3). The resulted estimator is called the MCC-KF method. Taking into account that only a finite number of data points k = 1, ... N is available in practice, the sample mean is utilized in corresponding formulas. The problem of estimating x_k for state-space model (1), (2) is equivalent to maximizing J(k) given by

$$\hat{x}_{k|k} = \arg\max J(k) \tag{7}$$

where

$$J(k) = k_{\sigma}(\|\hat{x}_{k|k} - F_{k-1}\hat{x}_{k-1|k-1}\|_{P_{k|k-1}^{-1}}) + k_{\sigma}(\|y_k - H_k\hat{x}_{k|k}\|_{R_k^{-1}})$$
(8)

with the Gaussian kernel functions defined as follows [11]:

$$k_{\sigma}(\|\hat{x}_{k|k} - F_{k-1}\hat{x}_{k-1|k-1}\|_{P_{k|k-1}^{-1}}) = \exp\left\{-\frac{\|\hat{x}_{k|k} - F_{k-1}\hat{x}_{k-1|k-1}\|_{P_{k|k-1}^{-1}}^{2}}{2\sigma^{2}}\right\}$$
$$k_{\sigma}(\|y_{k} - H_{k}\hat{x}_{k|k}\|_{R_{k}^{-1}}) = \exp\left\{-\frac{\|y_{k} - H_{k}\hat{x}_{k|k}\|_{R_{k}^{-1}}^{2}}{2\sigma^{2}}\right\}.$$

The optimization condition $\partial J(k)/\partial \hat{x}_{k|k} = 0$ yields the nonlinear equation that should be solved with respect to $\hat{x}_{k|k}$

$$\hat{x}_{k|k} = F_{k-1}\hat{x}_{k-1|k-1} + \frac{k_{\sigma}\left(\|y_{k} - H_{k}\hat{x}_{k|k}\|_{R_{k}^{-1}}\right)}{k_{\sigma}\left(\|\hat{x}_{k|k} - F_{k-1}\hat{x}_{k-1|k-1}\|_{P_{k|k-1}^{-1}}\right)} H_{k}^{T}(y_{k} - H_{k}\hat{x}_{k|k}).$$
(9)

In [8, 11], a fixed point rule (with one iterate) is utilized for solving (9) where the initial approximation $\hat{x}_{k|k}^{(0)}$ is set to $\hat{x}_{k|k-1}$, i.e. $\hat{x}_{k|k}^{(0)} = \hat{x}_{k|k-1}$ is substituted at the right-hand side of equation (9). Thus, we get

$$\hat{x}_{k|k} = F_{k-1}\hat{x}_{k-1|k-1} + \lambda_k H_k^T (y_k - H_k \hat{x}_{k|k-1})$$
(10)

where λ_k stands for

$$\lambda_{k} = \frac{k_{\sigma}(||y_{k} - H_{k}\hat{x}_{k|k-1}||_{R_{k}^{-1}})}{k_{\sigma}(||\hat{x}_{k|k-1} - F_{k-1}\hat{x}_{k-1|k-1}||_{P_{k|k-1}^{-1}})}.$$
(11)

Finally, the recursion for the state estimate in (10) is utilized with the KF-like estimation and the related error covariance propagation [11]. The resulted estimator is called the maximum correntropy criterion Kalman filter (MCC-KF) and summarized as follows [11, p. 503]. For readers' convenience, it is presented here in the form of Algorithm 1.

Algorithm 1. MCC-KF (original MCC-KF)

INITIALIZATION:(k = 0) $\hat{x}_{0|0} = \bar{x}_0$ and $P_{0|0} = \Pi_0$. Time Update: $(k = \overline{1, N})$

1
$$\hat{x}_{k|k-1} = F_{k-1}\hat{x}_{k-1|k-1};$$

2 $P_{k|k-1} = F_{k-1}P_{k-1|k-1}F_{k-1}^T + G_{k-1}Q_{k-1}G_{k-1}^T;$ MEASUREMENT UPDATE: $(k = \overline{1, N})$

3 Compute λ_k by formula (11);

4
$$K_{k} = \lambda_{k} \left(P_{k|k-1}^{-1} + \lambda_{k} H_{k}^{T} R_{k}^{-1} H_{k} \right)^{-1} H_{k}^{T} R_{k}^{-1};$$

5
$$P_{k|k} = (I - K_{k} H_{k}) P_{k|k-1} (I - K_{k} H_{k})^{T} + K_{k} R_{k} K_{k}^{T}$$

6
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_{k} (y_{k} - H_{k} \hat{x}_{k|k-1}).$$

It is worth noting here that the MCC-KF coincides with the classical KF when $\lambda_k = 1$. Thus, similar to the classical KF equations presented in [18, p. 128-129], the following formulas have been obtained for the MCC-KF method in [19, Lemma 1]:

$$K_k = \lambda_k P_{k|k-1} H_k^T \left(\lambda_k H_k P_{k|k-1} H_k^T + R_k \right)^{-1}$$
(12)

$$=\lambda_k P_{k|k} H_k^T R_k^{-1} \tag{13}$$

;

where $R_{e,k} := \lambda_k H_k P_{k|k-1} H_k^T + R_k$, and $P_{k|k}$ satisfies

$$P_{k|k} = \left(P_{k|k-1}^{-1} + \lambda_k H_k^T R_k^{-1} H_k\right)^{-1}$$
(14)

$$= (I - K_k H_k) P_{k|k-1}$$
(15)

$$= (I - K_k H_k) P_{k|k-1} (I - \lambda_k K_k H_k)^T + K_k R_k K_k^T.$$
(16)

It is not difficult to see that the gain matrix K_k in the MCC-KF implementation (see line 4 in Algorithm 1) is computed by formula (13) where the error covariance matrix $P_{k|k}$ obeys equation (14), i.e. the following formula holds: $P_{k|k}$ = $\lambda_k \left(P_{k|k-1}^{-1} + \lambda_k H_k^T R_k^{-1} H_k \right)^{-1} H_k^T R_k^{-1}$. Next, having compared equation (16) for computing $P_{k|k}$ with the equation in line 5 of Algorithm 1, we conclude that the suggested MCC-KF implementation (Algorithm 1) neglects the scalar parameter λ_k in (16) in order to keep the symmetric form. In the KF community, the symmetric equation (see line 5 of Algorithm 1) is called the Joseph stabilized form. It is recognized to be the preferable implementation strategy for the classical KF, because it ensures the symmetric form of error covariance matrix $P_{k|k}$ in the presence of roundoff errors and, hence, improves the numerical robustness [18, 20]. In summary, the MCC-KF implies the simplified error covariance computation strategy (λ_k is omitted) in order to keep the reliable Joseph stabilized form. However, a loss in estimation quality is the price to be paid; see the numerical results and the improved MCC-KF (IMCC-KF) variant developed in [19]. For readers' convenience, the IMCC-KF is summarized in Algorithm 2.

Algorithm 2. IMCC-KF (improved MCC-KF)

INITIALIZATION: $(k = 0) \hat{x}_{0|0} = \bar{x}_0$ and $P_{0|0} = \Pi_0$. TIME UPDATE: $(k = \overline{1, N})$ $\hat{x}_{k|k-1} = F_{k-1}\hat{x}_{k-1|k-1};$ $P_{k|k-1} = F_{k-1}P_{k-1|k-1}F_{k-1}^T + G_{k-1}Q_{k-1}G_{k-1}^T;$

MEASUREMENT UPDATE: $(k = \overline{1, N})$

- 3 Compute λ_k by formula (11);
- 4 $K_k = \lambda_k P_{k|k-1} H_k^T \left(\lambda_k H_k P_{k|k-1} H_k^T + R_k \right)^{-1};$

5
$$P_{k|k} = (I - K_k H_k) P_{k|k-1};$$

6 $\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - H_k \hat{x}_{k|k-1}).$

1

2

As can be seen, the IMCC-KF (Algorithm 2) implies equation (12) for the gain matrix K_k computation (line 4 in Algorithm 2) and formula (15) for the error covariance matrix $P_{k|k}$ calculation (line 5 in Algorithm 2). Both the MCC-KF and IMCC-KF are shown to outperform the classical KF for estimation accuracy in case of non-Gaussian uncertainties and outliers [8, 11, 19]. Meanwhile, to deal with the numerical instability problem and robustness with respect to roundoff errors, we derive the so-called *factored-form* implementations [21]: "It was recognized in the KF community that the factoredform (square-root) algorithms are the preferred implementations when a high operational reliability is required".

The key idea of the factored-form methodology for implementing linear and nonlinear KF-like estimators is to factorize the error covariance matrix P in the form of $P = SS^T$ and, then, re-formulate the underlying filtering equations in terms of these factors, only. Thus, the square-root algorithms recursively update the factors S instead of entire matrix P at each filtering step. It ensures the theoretical properties of any covariance matrix, i.e. its symmetric form and positive semi-definiteness, in the presence of roundoff errors. Indeed, when all measurements are processed, the full matrix P is re-constructed from the updated factors S by backward multiplication $SS^T = P$. Although, the roundoff errors influence the factors S, the product $SS^T = P$ is a symmetric and positive semi-definite matrix.

It is worth noting here that the factorization $P = SS^T$ can be implemented in various ways. This yields a variety of the factored-form (square-root) strategies: (i) Cholesky factorization-based algorithms, e.g. in [22, 23, 24]; (ii) UD-based implementations in [25], and (iii) SVD-based methods published recently in [26]. We stress that all cited square-root methods have been derived for the classical KF. Meanwhile for the MCC methodology the Cholesky-based IMCC-KF implementation has been proposed, only [19]. The goal of this paper is to proceed our recent research and suggest a complete factored-form family for both the MCC-KF (Algorithm 1) and IMCC-KF (Algorithm 2) estimators. Finally, it is important for further derivation that the scalar λ_k in the MCC-KF and IMCC-KF methods is a nonnegative value and, hence, a square root exists (for real nonnegative numbers).

2. The Cholesky factored-form implementations

The Cholesky factorization-based approach is the most popular strategy for designing factored-form implementations and, hence, traditionally used in engineering literature. It implies factorization of a symmetric positive definite matrix A in the form $A = (A^{1/2})(A^{1/2})^T$ where the factor $A^{1/2}$ is an upper or lower triangular matrix with positive diagonal elements. The resulted filtering methods belong to factored-form (square-root) family because the matrix square root S (i.e. $P = SS^T$) can be defined as $S := P^{1/2}$. All algorithms derived in this paper utilize the Cholesky decomposition in the form $A = A^{T/2}A^{1/2}$ where $A^{1/2}$ is an upper triangular matrix with positive diagonal elements¹.

Previously, the array Cholesky-based implementations have been developed for the IMCC-KF (Algorithm 2), only [19]. Thus, they are not presented here in details, but their key properties are discussed. The so-called array form is the preferable filtering implementation because of the following reasons [24]: (i) it makes algorithms convenient for practical use and suitable for parallel implementation; (ii) utilization of stable orthogonal transformation at each iteration step improves numerical stability. In summary, the array Cholesky-based filters utilize QR factorization for updating the corresponding Cholesky factors as follows: the filter quantities are compiled into the pre-array A and, next, an orthogonal operator \mathfrak{V} is applied $\mathfrak{V}A = \mathbb{R}$ in order to obtain the required triangular form of the post-array \mathbb{R} . The updated filter quantities are simply read-off from the post-array R. To summarize, the Cholesky-based IMCC-KF algorithm implies the following factorizations for updating $P_{k|k-1}^{1/2}$ and $P_{k|k}^{1/2}$ at the time and measurement steps [19, Algorithm 2]:

$$\mathfrak{B}\underbrace{\begin{bmatrix} P_{k-1|k-1}^{1/2} F_{k-1}^T\\ Q_{k-1}^{1/2} G_{k-1}^T \end{bmatrix}}_{\text{Pre-array }\mathbb{A}} = \underbrace{\begin{bmatrix} P_{k|k-1}^{1/2}\\ 0 \end{bmatrix}}_{\text{Post-array }\mathbb{R}}$$
(17)

$$\mathfrak{W}\underbrace{\begin{bmatrix} R_k^{1/2} & 0\\ \lambda_k^{1/2} P_{k|k-1}^{1/2} H_k^T & P_{k|k-1}^{1/2} \end{bmatrix}}_{\text{Pre-array }\mathbb{A}} = \underbrace{\begin{bmatrix} R_{e,k}^{1/2} & \bar{K}_k^T\\ 0 & P_{k|k}^{1/2} \end{bmatrix}}_{\text{Post-array }\mathbb{R}}$$
(18)

where $\bar{K}_k = \lambda_k^{-1/2} K_k R_{e,k}^{T/2} = \lambda_k^{1/2} P_{k|k-1} H_k^T R_{e,k}^{-1/2}$ is the "normalized" feedback gain and $R_{e,k} = \lambda_k H_k P_{k|k-1} H_k^T + R_k$.

Formulas (17), (18) are algebraic equivalent to corresponding equations in the conventional IMCC-KF implementation (Algorithm 2). It is not difficult to prove, if we note that the orthogonal transformations set up a conformal (i.e. a normand angle-preserving) mapping between the (block) columns of the pre-array \mathbb{A} and the columns of the post-array \mathbb{R} . More precisely, let us consider equation (18) in detail. Because of a norm-preserving mapping, the first inner product is

$$< [R_k^{1/2} \ \lambda_k^{1/2} P_{k|k-1}^{1/2} H_k^T], [R_k^{1/2} \ \lambda_k^{1/2} P_{k|k-1}^{1/2} H_k^T] > = < [X \ 0], [X \ 0] > .$$

Hence, we get $X^T X = \lambda_k H_k \underbrace{P_{k|k-1}^{T/2} P_{k|k-1}^{1/2}}_{P_{k|k-1}} H_k^T + \underbrace{R_k^{T/2} R_k^{1/2}}_{R_k} = R_{e,k},$

i.e. $X := R_{e,k}^{1/2}$. At the same way, we define

$$< [R_k^{1/2} \quad \lambda_k^{1/2} P_{k|k-1}^{1/2} H_k^T], [0 \quad P_{k|k-1}^{1/2}] > = < [R_{e,k}^{1/2} \quad 0], [Y \quad Z] >, \\ < [0 \quad P_{k|k-1}^{1/2}], [0 \quad P_{k|k-1}^{1/2}] > = < [Y \quad Z], [Y \quad Z] >$$

and obtain the set of equations

$$\lambda_k H_k P_{k|k-1} = R_{e,k}^{T/2} Y, \qquad P_{k|k-1} = Z^T Z + Y^T Y.$$

Thus, we get $Y := \lambda_k^{1/2} R_{e,k}^{-T/2} H_k P_{k|k-1} = \tilde{K}_k^T$ and, hence,

$$Z^{T}Z = P_{k|k-1} - Y^{T}Y = P_{k|k-1} - \bar{K}_{k}\bar{K}_{k}^{T}$$

= $P_{k|k-1} - \lambda_{k}P_{k|k-1}H_{k}^{T}R_{e,k}^{-1}H_{k}P_{k|k-1}$
= $P_{k|k-1} - K_{k}H_{k}P_{k|k-1} = (I - K_{k}H_{k})P_{k|k-1} = P_{k|k},$

¹Notation to be used: $A^{T/2} \equiv (A^{1/2})^T$, $A^{-1/2} \equiv (A^{1/2})^{-1}$, $A^{-T/2} \equiv (A^{-1/2})^T$.

i.e. from (15) we conclude $P_{k|k} = Z^T Z$ and, hence, $Z := P_{k|k}^{1/2}$.

Thus, factorization (18) implies formulas in lines 4 and 5 of Algorithm 2. Meanwhile equation (17) yields formula in line 2 of Algorithm 2. Indeed,

$$< [P_{k-1|k-1}^{1/2} F_{k-1}^T \quad Q_{k-1}^{1/2} G_{k-1}^T], [P_{k-1|k-1}^{1/2} F_{k-1}^T \quad Q_{k-1}^{1/2} G_{k-1}^T] >$$

= <[X 0], [X 0]>,

i.e. we get $X^T X = F_{k-1} P_{k-1|k-1} F_{k-1}^T + G_{k-1} Q_{k-1} G_{k-1}^T$ and, hence, we conclude $X := P_{k|k-1}^{1/2}$.

The array Cholesky-based MCC-KF implementation (Algorithm 1) can be derived at the same way. Indeed, the time update stage in Algorithms 1 and 2 is the same and, hence, these parts coincide in the Cholesky-based counterparts as well. We conclude that formula (17) holds for the MCC-KF. The difference is in the measurement update stage. Having analyzed the equation for gain K_k computation in line 4 of the MCC-KF (Algorithm 1), we conclude that the error covariance matrix is also computed at the same line. Indeed, $K_k = \lambda_k \left(P_{k|k-1}^{-1} + \lambda_k H_k^T R_k^{-1} H_k \right)^{-1} H_k^T R_k^{-1} \text{ in the MCC-KF (Al$ gorithm 1) where $(P_{k|k-1}^{-1} + \lambda_k H_k^T R_k^{-1} H_k)^{-1} = P_{k|k}$ according to equation (14). However, at the last line of Algorithm 1, matrix $P_{k|k}$ is re-computed by the Joseph stabilized form derived for the classical KF, i.e. with the skipped λ_k value; see formula (16) and the discussion in Section 1. This means that $P_{k|k}$ in the MCC-KF implementations should be always re-computed at the end. Besides, the equations for computing K_k and $P_{k|k}$ in Algorithm 1 should be processed separately. These formulas are incompatible for combining them into unique array (because of the skipped λ_k), in contrast to the IMCC-KF (Algorithm 2) and equation (18). This results into the following implementation.

Algorithm 1a. SR MCC-KF (Cholesky-based MCC-KF)

INITIALIZATION:(k = 0) $\hat{x}_{0|0} = \bar{x}_0$ and $P_{0|0}^{1/2} = \Pi_0^{1/2}$ where Cholesky decomposition is applied: $\Pi_0 = \Pi_0^{T/2} \Pi_0^{1/2}$. TIME UPDATE: $(k = \overline{1, N})$

1 $\hat{x}_{k|k-1} = F_{k-1}\hat{x}_{k-1|k-1};$

2

5

Build the pre-array and apply QR factorization:

$$\mathfrak{B} \underbrace{\begin{bmatrix} P_{k-1|k-1}^{1/2} F_{k-1}^T \\ Q_{k-1}^{1/2} G_{k-1}^T \end{bmatrix}}_{\text{Pre-array } \mathbb{A}} = \underbrace{\begin{bmatrix} P_{k|k-1}^{1/2} \\ 0 \end{bmatrix}}_{\text{Post-array } \mathbb{R}} \stackrel{\text{read-off}}{\Longrightarrow} \begin{bmatrix} P_{k|k-1}^{1/2} \end{bmatrix};$$

MEASUREMENT UPDATE: $(k = \overline{1, N})$

3 Compute
$$\lambda_k$$
 by formula (11):

$$\mathfrak{W}\underbrace{\begin{bmatrix}P_{k|k-1}^{-T/2}\\\lambda_{k}^{1/2}R_{k}^{-T/2}H_{k}\end{bmatrix}}_{\text{Pre-array }\mathbb{A}} = \underbrace{\begin{bmatrix}P_{k|k}^{-T/2}\\0\end{bmatrix}}_{\text{Post-array }\mathbb{R}} \stackrel{\text{read-off}}{\Longrightarrow} \begin{bmatrix}P_{k|k}^{-T/2}\end{bmatrix};$$

$$Compute K_{k} = \lambda_{k} \left([P_{k|k}^{-T/2}]^{T}[P_{k|k}^{-T/2}]\right)^{-1} H_{k}^{T}R_{k}^{-1}$$

6
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - H_k \hat{x}_{k|k-1}).$$

$$\mathfrak{Q}\underbrace{\begin{bmatrix}P_{k|k-1}^{P_{l}(I)}(I-K_{k}H_{k})^{T}\\R_{k}^{1/2}K_{k}^{T}\end{bmatrix}}_{\text{Pre-array }\mathbb{A}} = \underbrace{\begin{bmatrix}P_{k|k}^{P_{l}(I)}\\0\end{bmatrix}}_{\text{Post-array }\mathbb{R}} \stackrel{\text{read-off}}{\Longrightarrow} \begin{bmatrix}P_{k|k}^{1/2}\end{bmatrix}$$

Remark 1. The values appeared in square brackets in all algorithms in this paper denote the blocks that are directly read-off from the corresponding post-arrays.

Following our previous analysis, it is not difficult to validate formulas in lines 4 and 7 of Algorithm 1a. Indeed, the following set of equations holds:

$$< [P_{k|k-1}^{-T/2} \ \lambda_k^{1/2} R_k^{-T/2} H_k] > < [P_{k|k-1}^{-T/2} \ \lambda_k^{1/2} R_k^{-T/2} H_k] >$$

$$= < [X \ 0], [X \ 0] >,$$

$$< [P_{k|k-1}^{1/2} (I - K_k H_k)^T \ R_k^{1/2} K_k^T], [P_{k|k-1}^{1/2} (I - K_k H_k)^T \ R_k^{1/2} K_k^T] >$$

$$= < [Y \ 0], [Y \ 0] >,$$

i.e. $X^T X = P_{k|k-1}^{-1} + \lambda_k H_k^T R_k^{-1} H_k$ and $Y^T Y = (I - K_k H_k) P_{k|k-1} (I - K_k H_k)^T + K_k R_k K_k^T$. Taking into account formula (14), we conclude $X := P_{k|k}^{-T/2}$ and, next, $Y := P_{k|k}^{1/2}$.

As discussed above, although the inverse $P_{k|k}^{-1/2}$ is already available from line 4 of Algorithm 1a, we cannot avoid line 7 for computing $P_{k|k}^{1/2}$ according to the Joseph stabilized equation. In fact, if the second orthogonal transformation at the measurement update step in Algorithm 1a is skipped (i.e. we simply inverse the already computed $P_{k|k}^{-1/2}$ to obtain $P_{k|k}^{1/2}$), then the resulted algorithm is algebraic equivalent to the IMCC-KF (Algorithm 2), but not to the MCC-KF (Algorithm 1). Indeed, the matrix $P_{k|k}^{-1/2}$ calculation in line 4 is, in fact, formula (14) that is equivalent to equation (16), but not to the symmetric Joseph stabilized form of the classical KF recursion utilized in line 7 of the MCC-KF (Algorithm 1). Thus, to ensure algebraic equivalence between all MCC-KF implementations, the extra computations related to the symmetric classical KF formula for $P_{k|k}$ update are not avoidable and should be performed in any case.

3. The UD-based factored-form implementations

The first UD-based KF algorithms have been developed by Thornton and Bierman [27, 28]. The key idea of this strategy is to avoid square-rooting. Due to the computational complexity reasons, the square-root-free methods were preferable for practical implementations in 1970s; see also filtering methods in [29]. For modern computational devices the square root operation is not a problem because it can be implemented efficiently. However, the UD-based filters still deserve some merit. One of possible reasons is utilization of square-root-free orthogonal rotations that might be more numerically stable than usual QR decomposition; see the discussion in [30, 31, 32, 33]. Thereby, the UD-based estimators' quality and robustness are enhanced in ill-conditioned situations [26, 28]. It is also worth noting here that the first UD-based KF implementations were derived in sequential form, i.e. when the available measurement vector y_k is processed in a component-wise manner. Nowadays, the advantageous array form is preferable for practical implementation. For the classical KF, such array implementations have been suggested in [20, 34] as well as the extended array algorithms have been derived in [35, 36]. In this section, the array UD-based methods are derived for both the MCC-KF (Algorithm 1) and IMCC-KF (Algorithm 2) estimators.

Consider the modified Cholesky decomposition $P = \bar{U}_P D_P \bar{U}_P^T$ where D_P denotes a diagonal matrix and \bar{U}_P is an upper triangular matrix with 1's on the main diagonal [25]. The UD-based implementations belong to factored-form (square-root) family because the matrix square root *S* (i.e. $P = SS^T$) can be defined as $S := \bar{U}_P D_P^{1/2}$. As usual, the underlying filter recursion should be re-formulated for updating the resulted \bar{U}_P and D_P factors instead of full matrix *P*. In this paper, the modified weighted Gram-Schmidt (MWGS) orthogonalization is utilized for updating the resulted UD factors as follows [25, Lemma VI.4.1]: given $\mathbb{A} \in \mathbb{R}^{r \times s}$, $r \geq s$ and diagonal $\mathbb{D}_A \in \mathbb{R}^{s \times s}$ and diagonal matrix $\mathbb{D}_B \in \mathbb{R}^{s \times s}$, i.e.

$$\mathbb{A} = \mathfrak{W}\mathbb{B}^T \quad \text{with} \quad \mathfrak{W}^T \mathbb{D}_A \mathfrak{W} = \mathbb{D}_B \tag{19}$$

where $\mathfrak{W} \in \mathbb{R}^{r \times s}$ is the MWGS orthogonalization.

Taking into account properties of orthogonal matrices, from equation (19) we obtain

$$\mathbb{A}^T \mathbb{D}_A \mathbb{A} = \mathbb{B} \mathfrak{W}^T \mathbb{D}_A \mathfrak{W} \mathbb{B}^T = \mathbb{B} \mathbb{D}_B \mathbb{B}^T.$$
(20)

The first equation in (19) can be re-written as follows: $\mathbb{A}^T = \mathbb{B}\mathfrak{W}^T$, i.e. the orthogonal transformation sets up a conformal mapping between the (block) rows of the pre-array \mathbb{A}^T and the rows of the post-array \mathbb{B} where a diagonally weighted norms are utilized and preserved. Thus, we derive the following UD-based implementations.

Algorithm 1b. UD MCC-KF (UD-based MCC-KF)

INITIALIZATION:(k = 0) $\hat{x}_{0|0} = \bar{x}_0$ and $\bar{U}_{P_{0|0}} = \bar{U}_{\Pi_0}$, $D_{P_{0|0}} = D_{\Pi_0}$ where UD-decomposition is applied: $\Pi_0 = \bar{U}_{\Pi_0} D_{\Pi_0} \bar{U}_{\Pi_0}^T$. TIME UPDATE: $(k = \overline{1, N})$

1 $\hat{x}_{k|k-1} = F_{k-1}\hat{x}_{k-1|k-1};$

2 Build pre-arrays \mathbb{A} , \mathbb{D}_A and apply MWGS algorithm:

$$\underbrace{\begin{bmatrix} F_{k-1}\bar{U}_{P_{k-1|k-1}} & G_{k-1}\bar{U}_{Q_{k-1}} \end{bmatrix}}_{\text{Pre-array }\mathbb{A}^T} = \underbrace{\begin{bmatrix} \bar{U}_{P_{k|k-1}} \end{bmatrix}}_{\text{Post-array }\mathbb{B}} \mathfrak{B}^T \stackrel{\text{read-off}}{\Longrightarrow} \begin{bmatrix} \bar{U}_{P_{k|k-1}} \end{bmatrix};$$

$$\mathfrak{B}^T \underbrace{\begin{bmatrix} \text{diag } \{ D_{P_{k-1|k-1}}, D_{Q_{k-1}} \} \end{bmatrix}}_{\text{Pre-array }\mathbb{D}_A} \mathfrak{B} = \underbrace{\begin{bmatrix} D_{P_{k|k-1}} \end{bmatrix}}_{\text{Post-array }\mathbb{D}_B} \stackrel{\text{read-off}}{\Longrightarrow} \begin{bmatrix} D_{P_{k|k-1}} \end{bmatrix};$$

MEASUREMENT UPDATE: $(k = \overline{1, N})$

3 Compute λ_k by formula (11);

4 Build pre-arrays
$$\mathbb{A}$$
, \mathbb{D}_A and apply MWGS algorithm:

$$\underbrace{\begin{bmatrix} \bar{U}_{P_{k|k-1}}^{-T} & \lambda_k^{1/2} H_k^T \bar{U}_{R_k}^{-T} \end{bmatrix}}_{\text{Pre-array } \mathbb{A}^T} = \underbrace{\begin{bmatrix} \bar{U}_{P_{k|k}}^{-T} \end{bmatrix}}_{\text{Post-array } \mathbb{B}} \mathfrak{W}^T \stackrel{\text{read-off}}{\Longrightarrow} \begin{bmatrix} \bar{U}_{P_{k|k}}^{-T} \end{bmatrix};$$
$$\mathfrak{W}^T \underbrace{\begin{bmatrix} \text{diag} \left\{ D_{P_{k|k-1}}^{-1}, D_{R_k}^{-1} \right\} \right]}_{\text{Pre-array } \mathbb{D}_A} \mathfrak{W} = \underbrace{\begin{bmatrix} D_{P_{k|k}}^{-1} \end{bmatrix}}_{\text{Post-array } \mathbb{D}_B} \stackrel{\text{read-off}}{\Longrightarrow} \begin{bmatrix} D_{P_{k|k}}^{-1} \end{bmatrix};$$

Compute
$$K_k = \lambda_k \left([\bar{U}_{P_{k|k}}^{-T}] [D_{P_{k|k}}^{-1}] [\bar{U}_{P_{k|k}}^{-T}]^T \right)^{-1} H_k^T R_k^{-1};$$

6 $\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - H_k \hat{x}_{k|k-1}).$

5

7

Build pre-arrays
$$\mathbb{A}$$
, \mathbb{D}_A and apply MWGS algorithm:

$$\underbrace{\left[(I - K_k H_k) \overline{U}_{P_{k|k-1}} \quad K_k \overline{U}_{R_k}\right]}_{\text{Pre-array } \mathbb{A}^T} \stackrel{\text{read-off}}{\Longrightarrow} \begin{bmatrix} \overline{U}_{P_{k|k}} \end{bmatrix};$$

$$\mathfrak{Q}^T \underbrace{\left[\text{diag}\left\{D_{P_{k|k-1}}, D_{R_k}\right\}\right]}_{\text{Pre-array } \mathbb{D}_A} \mathfrak{Q} \stackrel{\text{ead-off}}{\Longrightarrow} \begin{bmatrix} D_{P_{k|k}} \end{bmatrix}.$$

To validate the algorithm above, we start with the MWGS orthogonalization in line 2 of Algorithm 1b, i.e.

$$< [F_{k-1}\bar{U}_{P_{k-1|k-1}} \quad G_{k-1}\bar{U}_{Q_{k-1}}] > < [F_{k-1}\bar{U}_{P_{k-1|k-1}} \quad G_{k-1}\bar{U}_{Q_{k-1}}] >_{\mathbb{D}_{A}} \\ = X\mathbb{D}_{X}X^{T} \quad \text{where} \quad \mathbb{D}_{A} = \text{diag}\left\{D_{P_{k-1|k-1}}, D_{Q_{k-1}}\right\},$$

i.e. the following equation holds

$$X\mathbb{D}_{X}X^{T} = F_{k-1}\underbrace{\bar{U}_{P_{k-1|k-1}}D_{P_{k-1|k-1}}\bar{U}_{P_{k-1|k-1}}^{T}}_{P_{k-1|k-1}}F_{k-1}^{T}$$
$$+ G_{k-1}(\bar{U}_{Q_{k-1}}D_{Q_{k-1}}\bar{U}_{Q_{k-1}}^{T})G_{k-1}^{T} = P_{k|k-1}$$

where the process covariance Q_{k-1} is UD-factorized as well, i.e. $Q_{k-1} = \overline{U}_{Q_{k-1}} D_{Q_{k-1}} \overline{U}_{Q_{k-1}}^T$. Hence, we conclude $X := \overline{U}_{P_{k|k-1}}$ and $\mathbb{D}_X := D_{P_{k|k-1}}$.

At the same manner, the formulas in lines 4 and 7 of Algorithm 1b are validated, i.e.

$$< [\bar{U}_{P_{k|k-1}}^{-T} \quad \lambda_{k}^{1/2} H_{k}^{T} \bar{U}_{R_{k}}^{-T}] > < [\bar{U}_{P_{k|k-1}}^{-T} \quad \lambda_{k}^{1/2} H_{k}^{T} \bar{U}_{R_{k}}^{-T}] >_{\mathbb{D}_{A}}$$

$$= Y \mathbb{D}_{Y} Y^{T} \quad \text{where} \quad \mathbb{D}_{A} = \text{diag} \left\{ D_{P_{k|k-1}}^{-1}, D_{R_{k}}^{-1} \right\},$$

$$< [(I - K_{k} H_{k}) \bar{U}_{P_{k|k-1}} \quad K_{k} \bar{U}_{R_{k}}], [(I - K_{k} H_{k}) \bar{U}_{P_{k|k-1}} \quad K_{k} \bar{U}_{R_{k}}] >$$

$$= Z \mathbb{D}_{Z} Z^{T} \quad \text{where} \quad \mathbb{D}_{A} = \text{diag} \left\{ D_{P_{k|k-1}}, D_{R_{k}} \right\}$$

and the measurement covariance matrix R_k is UD-factorized, i.e. $R_k = \bar{U}_{R_k} D_{R_k} \bar{U}_{R_k}^T$. Thus, we get

$$Y\mathbb{D}_{Y}Y^{T} = \underbrace{\bar{U}_{P_{k|k-1}}^{-T}D_{P_{k|k-1}}^{-1}\bar{U}_{P_{k|k-1}}^{-1}}_{P_{k|k-1}^{-1}} + \lambda_{k}H_{k}^{T}\underbrace{\bar{U}_{R_{k}}^{-T}D_{R_{k}}^{-1}\bar{U}_{R_{k}}^{-1}}_{R_{k}^{-1}}H_{k}.$$

Having compared equation above with formula (14), we conclude that $Y := \overline{U}_{P_{kk}}^{-T}$ and $\mathbb{D}_Y := D_{P_{kk}}^{-1}$. Next,

$$Z\mathbb{D}_{Z}Z^{T} = (I - K_{k}H_{k})\underbrace{\bar{U}_{P_{k|k-1}}D_{P_{k|k-1}}\bar{U}_{P_{k|k-1}}^{T}}_{P_{k|k-1}}(I - K_{k}H_{k})^{T} + K_{k}(\bar{U}_{R_{k}}D_{R_{k}}\bar{U}_{R_{k}}^{T})K_{k}^{T} = P_{k|k}$$

and, hence, $Z := \overline{U}_{P_{k|k}}$ and $\mathbb{D}_Z := D_{P_{k|k}}$. This completes the proof of algebraic equivalence between the original MCC-KF (Algorithm 1) and its UD-based counterpart (Algorithm 1b).

Finally, we suggest the UD-based IMCC-KF implementation in Algorithm 2. In fact, it requires one less MWGS orthogonalization at each step because the re-calculation of $P_{k|k}$ via the Joseph stabilized equation is not required in the IMCC-KF.

Algorithm 2b. UD IMCC-KF (UD-based IMCC-KF)

- INITIALIZATION: (k = 0) $\hat{x}_{0|0} = \bar{x}_0$ and $\bar{U}_{P_{0|0}} = \bar{U}_{\Pi_0}$, $D_{P_{0|0}} = D_{\Pi_0}$ where UD-decomposition is applied: $\Pi_0 = \bar{U}_{\Pi_0} D_{\Pi_0} \bar{U}_{\Pi_0}^T$. TIME UPDATE: $(k = \overline{1, N})$ Repeat lines 1,2 of Algorithm 1b; MEASUREMENT UPDATE: $(k = \overline{1, N})$
- 1 Compute λ_k by formula (11);
- 2 Build pre-arrays \mathbb{A} , \mathbb{D}_A and apply MWGS algorithm:

$$\underbrace{\begin{bmatrix} \bar{U}_{P_{k|k-1}} & 0\\ \lambda_k^{1/2} H_k \bar{U}_{P_{k|k-1}} & \bar{U}_{R_k} \end{bmatrix}}_{\text{Pre-array } \mathbb{A}^T} = \underbrace{\begin{bmatrix} \bar{U}_{P_{k|k}} & \bar{K}_k^u\\ 0 & \bar{U}_{R_{e,k}} \end{bmatrix}}_{\text{Post-array } \mathbb{B}} \mathfrak{D}^T \overset{\text{read-off}}{\Longrightarrow} \begin{bmatrix} \bar{U}_{P_{k|k}} \end{bmatrix}; \\ \mathfrak{D}_{R_{e,k}} \end{bmatrix}, [\bar{K}_k^u]; \\ \mathfrak{D}_{R_{e,k}}^T \underbrace{\begin{bmatrix} \text{diag} \{ D_{P_{k|k-1}}, D_{R_k} \} \}}_{\text{Pre-array } \mathbb{D}_A} \underbrace{\end{bmatrix}} \mathfrak{D} = \underbrace{\text{diag} \{ D_{P_{k|k}}, D_{R_{e,k}} \}}_{\text{Post-array } \mathbb{D}_B} \overset{\text{read-off}}{\Longrightarrow} \begin{bmatrix} D_{P_{k|k}} \end{bmatrix}; \\ \mathfrak{D}_{k|k} = \hat{x}_{k|k-1} + \lambda_k^{1/2} \begin{bmatrix} \bar{K}_k^u \end{bmatrix} \begin{bmatrix} \bar{U}_{R_{e,k}} \end{bmatrix}^{-1} (y_k - H\hat{x}_{k|k-1}).$$

3

The method in Algorithm 2b can be validated at the same manner as Algorithm 1b. More precisely, the time update step of these implementations is the same and, hence, it has been already proved. Let's consider the transformation in line 2 of Algorithm 2b. We have the following set of equations

$$< [\bar{U}_{P_{k|k-1}} \quad 0] > < [\bar{U}_{P_{k|k-1}} \quad 0] >_{\text{diag}\left\{D_{P_{k|k-1}}, D_{R_{k}}\right\}}$$

$$= <[X \quad Y], [X \quad Y] >_{\text{diag}\{D_{1}, D_{2}\}},$$

$$< [\bar{U}_{P_{k|k-1}} \quad 0] > < [\lambda_{k}^{1/2}H_{k}\bar{U}_{P_{k|k-1}} \quad \bar{U}_{R_{k}}] >_{\text{diag}\left\{D_{P_{k|k-1}}, D_{R_{k}}\right\}}$$

$$= <[X \quad Y], [0 \quad Z] >_{\text{diag}\{D_{1}, D_{2}\}},$$

$$< [\lambda_{k}^{1/2}H_{k}\bar{U}_{P_{k|k-1}} \quad \bar{U}_{R_{k}}], [\lambda_{k}^{1/2}H_{k}\bar{U}_{P_{k|k-1}} \quad \bar{U}_{R_{k}}] >_{\text{diag}\left\{D_{P_{k|k-1}}, D_{R_{k}}\right\}}$$

$$= <[0 \quad Z], [0 \quad Z] >_{\text{diag}\{D_{1}, D_{2}\}}.$$

From the last equation, we have

$$ZD_2Z^T = \lambda_k H_k \underbrace{(\overline{U}_{P_{k|k-1}}D_{P_{k|k-1}}\overline{U}_{P_{k|k-1}}^T)}_{P_{k|k-1}}H_k^T + \underbrace{(\overline{U}_{R_k}D_{R_k}\overline{U}_{R_k}^T)}_{R_k} = R_{e,k},$$

i.e. $Z := \bar{U}_{R_{e,k}}$ and $D_2 := D_{R_{e,k}}$.

Having substituted the resulted Z and D_2 values into the second equation in the set above, we obtain

$$YD_{R_{e,k}}\bar{U}_{R_{e,k}}^{T} = \lambda_{k}^{1/2}(\bar{U}_{P_{k|k-1}}D_{P_{k|k-1}}\bar{U}_{P_{k|k-1}}^{T})H_{k}^{T}$$

Hence, $Y := \lambda_k^{1/2} P_{k|k-1} H_k^T \bar{U}_{R_{ek}}^{-T} D_{R_{ek}}^{-1}$. This value is denoted as \bar{K}_k^u in Algorithm 2b, i.e. $Y := \bar{K}_k^u$. From formula (12), the following relationship is obtained: $K_k = \lambda_k P_{k|k-1} H_k^T R_{e,k}^{-1} = \lambda_k^{1/2} \bar{K}_k^u \bar{U}_{R_{ek}}^{-1}$. Next, we note that the block \bar{K}_k^u is directly read-off from post-array in Algorithm 2b. Hence, it makes sense to use it for computing state estimate, straightforward

$$\begin{split} \hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_k(y_k - H_k \hat{x}_{k|k-1}) \\ &= \hat{x}_{k|k-1} + \lambda_k^{1/2} \bar{K}_k^u \bar{U}_{R_{e,k}}^{-1}(y_k - H_k \hat{x}_{k|k-1}), \end{split}$$

i.e. the formula in line 3 of Algorithm 2b is validated.

Finally, we consider the last relationship that is

$$XD_1X^T + \bar{K}_k^u D_{R_{e,k}} (\bar{K}_k^u)^T = P_{k|k-1},$$

i.e. we have

$$\begin{aligned} XD_1 X^T &= P_{k|k-1} - \lambda_k^{1/2} P_{k|k-1} H_k^T \bar{U}_{R_{e,k}}^{-T} D_{R_{e,k}}^{-1} D_{R_{e,k}} (\lambda_k^{-1/2} K_k \bar{U}_{R_{e,k}})^T \\ &= P_{k|k-1} - P_{k|k-1} H_k^T K_k^T = P_{k|k-1} (I - H_k^T K_k^T) = P_{k|k}^T. \end{aligned}$$

Taking into account a symmetric form of any covariance matrix, we conclude $X := \overline{U}_{P_{k|k}}$ and $D_1 := D_{P_{k|k}}$. This completes the proof of the UD-based IMCC-KF (Algorithm 2b).

4. SVD-based factored-form implementations

To the best of author's knowledge, there exist only two *classical* KF methods based on the singular value decomposition (SVD). The first SVD-based KF was developed in [37]. However, it was shown to be numerically unstable with respect to

roundoff errors. Thereby, the robust SVD-based KF algorithm has been recently proposed in [26]. In this paper, the goal is to extend the SVD-based filtering on the MCC-KF (Algorithm 1) and IMCC-KF (Algorithm 2) techniques.

Each iteration of the new filtering methods is implemented by using the SVD factorization [38, Theorem 1.1.6]: Every matrix $A \in \mathbb{C}^{m \times n}$ of rank *r* can be written as follows:

$$A = W\Sigma V^*, \Sigma = \begin{bmatrix} S & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{m \times n}, S = \text{diag}\{\sigma_1, \dots, \sigma_r\}$$

where $W \in \mathbb{C}^{m \times m}$, $V \in \mathbb{C}^{n \times n}$ are unitary matrices, V^* is the conjugate transpose of V, and $S \in \mathbb{R}^{r \times r}$ is a real nonnegative diagonal matrix. Here $\sigma_1 \ge \sigma_2 \ge \ldots \ge \sigma_r > 0$ are called the singular values of A. (Note that if r = n and/or r = m, some of the zero submatrices in Σ are empty.)

The SVD-based filtering methods belong to the *factored-form* (square-root) family, because the covariance *P* is factorized in the form $P = VDV^T$ where *V* is an orthogonal matrix and *D* is a diagonal matrix with singular values of *P*. Hence, one can set the matrix square root as follows: $P = SS^T$ where $S := VD^{1/2}$. It is also worth noting here that in the SVD-based filtering, the square-root factor *S* is a full matrix, in general. This is in contrast to the upper or lower triangular factor *S* in the Cholesky-based implementations discussed in previous sections. Additionally, the SVD factorization provides the users with extra information about the matrix structure and properties and, hence, it might be used in various reduced-rank filtering design strategies.

The following SVD-based variant for the MCC-KF estimator (Algorithm 1) is proposed.

Algorithm 1c. SVD MCC-KF (SVD-based MCC-KF)

INITIALIZATION:(k = 0) $\hat{x}_{0|0} = \bar{x}_0$ and $V_{P_{0|0}} = V_{\Pi_0}$, $D_{P_{0|0}}^{1/2} = D_{\Pi_0}^{1/2}$ where SVD-decomposition is applied: $\Pi_0 = V_{\Pi_0} D_{\Pi_0} V_{\Pi_0}^T$. TIME UPDATE: $(k = \overline{1, N})$

1
$$\hat{x}_{k|k-1} = F_{k-1}\hat{x}_{k-1|k-1};$$

2 Build pre-array \mathbb{A} and apply SVD factorization

$$\underbrace{\begin{bmatrix} D_{P_{k-1|k-1}}^{1/2} V_{P_{k-1|k-1}}^T F_{k-1}^T \\ D_{Q_{k-1}}^{1/2} V_{Q_{k-1}}^T G_{k-1}^T \end{bmatrix}}_{\text{Pre-array }\mathbb{A}} = \underbrace{\mathfrak{M}}_{\text{Post-arrays}} \begin{bmatrix} D_{P_{k|k-1}}^{1/2} \\ \mathbb{B} \end{bmatrix}_{\text{Post-arrays}}^{\text{read-off}} \begin{bmatrix} D_{P_{k|k-1}}^{1/2} \\ \mathbb{B} \end{bmatrix};$$
MEASUREMENT UPDATE: $(k = \overline{1, N})$

3 Compute λ_k by formula (11);

Build pre-array
$$\mathbb{A}$$
 and apply SVD factorization

$$\begin{bmatrix} \lambda_k^{1/2} D_{R_k}^{-1/2} V_{R_k}^T H_k V_{P_{k|k-1}} \\ D_{P_{k|k-1}}^{-1/2} \end{bmatrix} = \mathfrak{M} \begin{bmatrix} D_{P_{k|k}}^{-1/2} \\ 0 \end{bmatrix} \mathfrak{B}^T \stackrel{\text{read-off}}{\Longrightarrow} \begin{bmatrix} D_{P_{k|k}}^{-1/2} \\ 0 \end{bmatrix};$$
Prove array \mathbb{A}

Calculate the SVD factor $V_{P_{k|k}} = V_{P_{k|k-1}}\mathfrak{B};$

Compute
$$K_k = \lambda_k [V_{P_{k|k}}] [D_{P_{k|k}}^{-1/2}]^{-2} [V_{P_{k|k}}]^T H_k^T R_k^{-1};$$

7 $\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - H_k \hat{x}_{k|k-1}).$

Build pre-array
$$\mathbb{A}$$
 and apply SVD factorization

$$\underbrace{\begin{bmatrix} D_{P_{k|k-1}}^{1/2} V_{P_{k|k-1}}^{T} (I - K_k H_k)^{T} \\ D_{R_k}^{1/2} V_{R_k}^{T} K_k^{T} \end{bmatrix}}_{\text{Pre-array } \mathbb{A}} = \underbrace{\mathfrak{W} \begin{bmatrix} D_{P_{k|k}}^{1/2} \\ 0 \end{bmatrix}}_{\text{Post-arrays}} \underbrace{\mathfrak{W} \xrightarrow{\text{read-off}}_{\text{Post-arrays}}}_{\text{Post-arrays}} \begin{bmatrix} D_{P_{k|k}}^{1/2} \\ 0 \end{bmatrix};$$

4

5 6

8

Remark 2. In the algorithm above, the SVD is applied to the process and measurement noise covariances as well, i.e $Q_k = V_{Q_k} D_{Q_k} V_{Q_k}^T$ and $R_k = V_{R_k} D_{R_k} V_{R_k}^T$. However, as discussed in [26, Remark 2], the Cholesky decomposition might be used for Q_k and R_k instead of the SVD factorization.

To prove the algebraic equivalence between the new SVDbased implementation (Algorithm 1c) and the original MCC-KF (Algorithm 1), we note that $A^T A = (V\Sigma W^T)(W\Sigma V^T)^T = V\Sigma^2 V^T$. Having compared both sides of the resulted equality $A^T A = V\Sigma^2 V^T$, one may validate the formulas in Algorithm 1c. Indeed, from the factorization in line 2, we obtain

$$F_{k-1}\underbrace{V_{P_{k-1|k-1}}D_{P_{k-1|k-1}}V_{P_{k-1|k-1}}^{T}}_{P_{k-1|k-1}}F_{k-1}^{T}+G_{k-1}\underbrace{V_{Q_{k-1}}D_{Q_{k-1}}V_{Q_{k-1}}^{T}}_{Q_{k-1}}G_{k-1}^{T}$$

$$=\mathfrak{B}D_{P_{k|k-1}}\mathfrak{B}^{T}=P_{k|k-1}, \text{ i.e. }\mathfrak{B}:=V_{P_{k|k-1}}.$$

Next, in line 4 of Algorithm 1c we have

$$\lambda_{k} V_{P_{k|k-1}}^{T} H_{k}^{T} \underbrace{V_{R_{k}} D_{R_{k}}^{-1} V_{R_{k}}^{T}}_{R_{k}^{-1}} H_{k} V_{P_{k|k-1}} + D_{P_{k|k-1}}^{-1} = \mathfrak{V} D_{P_{k|k}}^{-1} \mathfrak{V}^{T}.$$
(21)

Having multiplied both sides of equation (21) by $V_{P_{k|k-1}}$ (at the left) and by $V_{P_{k|k-1}}^T$ (at the right), we get

$$\lambda_{k}H_{k}^{T}R_{k}^{-1}H_{k} + \underbrace{V_{P_{k|k-1}}D_{P_{k|k-1}}^{-1}V_{P_{k|k-1}}^{T}}_{P_{k|k-1}^{-1}} = V_{P_{k|k-1}}\mathfrak{B}D_{P_{k|k}}^{-1}\mathfrak{B}^{T}V_{P_{k|k-1}}^{T}.$$
 (22)

Having compared formula (22) with equation (14), i.e. $P_{k|k} = (P_{k|k-1}^{-1} + \lambda_k H_k^T R_k^{-1} H_k)^{-1}$, we conclude that $V_{P_{k|k}} = V_{P_{k|k-1}} \mathfrak{V}$. This validates formula in line 5 of Algorithm 1c.

Next, formula for computing the gain K_k in line 6 is the same as in the original MCC-KF (Algorithm 1) where the SVD is used for matrix $P_{k|k}$. Finally, factorization in line 8 of Algorithm 1c implies the symmetric Joseph stabilized equation of the classical KF utilized in the MCC-KF (Algorithm 1), i.e.

$$A^{T}A = K_{k}R_{k}K_{k}^{T} + (I - K_{k}H_{k})\underbrace{V_{P_{k|k-1}}D_{P_{k|k-1}}V_{P_{k|k-1}}^{T}}_{P_{k|k-1}}(I - K_{k}H_{k})^{T}$$
$$= \mathfrak{B}D_{P_{k|k}}\mathfrak{B}^{T} = P_{k|k}, \text{ i.e. } \mathfrak{B} := V_{P_{k|k}}.$$

This concludes the proof of Algorithm 1c.

Similar, the SVD-based IMCC-KF is derived and summarized in Algorithm 2.

Algorithm 2c. SVD IMCC-KF (SVD-based IMCC-KF)

- INITIALIZATION: (k = 0) $\hat{x}_{0|0} = \bar{x}_0$ and $V_{P_{0|0}} = V_{\Pi_0}$, $D_{P_{0|0}}^{1/2} = D_{\Pi_0}^{1/2}$ where SVD-decomposition is applied: $\Pi_0 = V_{\Pi_0} D_{\Pi_0} V_{\Pi_0}^T$. TIME UPDATE: $(k = \overline{1, N})$ Repeat lines 1,2 of Algorithm 1c; MEASUREMENT UPDATE: $(k = \overline{1, N})$
- 1 Compute λ_k by formula (11);

2 Build pre-array
$$\mathbb{A}$$
 and apply SVD factorization

$$\underbrace{\begin{bmatrix} \lambda_k^{1/2} D_{R_k}^{-1/2} V_{R_k}^T H_k V_{P_{k|k-1}} \\ D_{P_{k|k-1}}^{-1/2} \end{bmatrix}}_{\text{Pre-array }\mathbb{A}} = \underbrace{\mathfrak{W} \begin{bmatrix} D_{P_{k|k}}^{-1/2} \\ 0 \end{bmatrix}}_{\text{Post-arrays}} \underbrace{\mathfrak{W} T}_{\text{Post-arrays}} \stackrel{\text{read-off}}{\Longrightarrow} \begin{bmatrix} D_{P_{k|k}}^{-1/2} \end{bmatrix};$$

3 Calculate the SVD factor $V_{P_{klk}} = V_{P_{klk-1}}\mathfrak{V};$

4 Compute
$$K_k = \lambda_k [V_{P_{k|k}}] [D_{P_{k|k}}^{-1/2}]^{-2} [V_{P_{k|k}}]^T H_k^T R_k^{-1};$$

5
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - H_k \hat{x}_{k|k-1}).$$

As in all previous methods, the time update steps in Algorithms 1c and 2c coincide. Next, the IMCC-KF requires one less SVD factorization at each filtering step because it does not demand the re-computation of $P_{k|k}$ by the symmetric Joseph stabilized equation at the end of each iterate. Thus, the IMCC-KF implementation (Algorithm 2c) is, in fact, the MCC-KF Algorithm 1c without the last $P_{k|k}$ re-calculation, i.e. all formulas of Algorithm 2c have been already proved in this section.

Our final remark concerns the robust variant of the SVDbased implementations. Both Algorithm 1c and Algorithm 2c require two matrix inversions that are related to $D_{P_{kk-1}}^{-1}$ and $D_{P_{kk}}^{-1}$ calculation. For numerical stability and computational complexity reasons, it is preferable to avoid this operation. Following [26], we can suggest the robust SVD-based implementation for the MCC-KF as follows.

Algorithm 1d. RSVD MCC-KF (robust SVD-based MCC-KF)

INITIALIZATION: (k = 0) $\hat{x}_{0|0} = \bar{x}_0$ and $V_{P_{0|0}} = V_{\Pi_0}$, $D_{P_{0|0}}^{1/2} = D_{\Pi_0}^{1/2}$ where SVD-decomposition is applied: $\Pi_0 = V_{\Pi_0} D_{\Pi_0} V_{\Pi_0}^T$. TIME UPDATE: $(k = \overline{1, N})$ Repeat lines 1,2 of Algorithm 1c; MEASUREMENT UPDATE: $(k = \overline{1, N})$

Compute λ_k by formula (11);

1

2

3 4

Build pre-array
$$\mathbb{A}$$
 and apply SVD factorization

$$\begin{bmatrix}
\lambda_{k}^{1/2} D_{P_{k|k-1}}^{1/2} V_{P_{k|k-1}}^{T} H_{k}^{T} \\
D_{R_{k}}^{1/2} V_{R_{k}}^{T}
\end{bmatrix} = \underbrace{\mathfrak{W} \begin{bmatrix} D_{R_{e,k}}^{1/2} \\ 0 \end{bmatrix}}_{\text{Post-arrays}} \underbrace{\mathbb{E} \begin{bmatrix} D_{R_{e,k}}^{1/2} \\ 0 \end{bmatrix}}_{\text{Post-arrays}} \begin{bmatrix} V_{R_{e,k}} = \mathfrak{B} \end{bmatrix};$$
Compute $K_{k} = \lambda_{k} P_{k|k-1} H_{k}^{T} [V_{R_{e,k}}] [D_{R_{e,k}}^{1/2}]^{-2} [V_{R_{e,k}}]^{T};$

5 Build pre-array
$$\mathbb{A}$$
 and apply SVD factorization

$$\underbrace{\begin{bmatrix} D_{P_{k|k-1}}^{1/2} V_{P_{k|k-1}}^T (I - K_k H_k)^T \\ D_{R_k}^{1/2} V_{R_k}^T K_k^T \end{bmatrix}}_{\text{Pre-array }\mathbb{A}} = \underbrace{\mathfrak{W} \begin{bmatrix} D_{P_{k|k}}^{1/2} \\ 0 \end{bmatrix}}_{\text{Post-arrays}} \underbrace{\mathfrak{W} \begin{bmatrix} D_{P_{k|k}}^{1/2} \\ 0 \end{bmatrix}}_{\text{Post-arrays}} \Longrightarrow [V_{P_{k|k}} = \mathfrak{V}].$$

The difference between Algorithms 1c and 1d is in the gain matrix K_k calculation. Algorithm 1c utilizes equation (13) while Algorithm 1d implies formula (12), i.e. $K_k = \lambda_k P_{k|k-1}H_k^T (\lambda_k H_k P_{k|k-1}H_k^T + R_k)^{-1}$. The matrix that needs to be inverted is denoted as $R_{e,k}$, i.e. $R_{e,k} = \lambda_k H_k P_{k|k-1} H_k^T + R_k$. Its SVD factors are computed in line 2 of Algorithm 1d, i.e.

$$A^{T}A = \lambda_{k}H_{k}\underbrace{V_{P_{k|k-1}}D_{P_{k|k-1}}V_{P_{k|k-1}}^{T}}_{P_{k|k-1}}H_{k}^{T} + \underbrace{V_{R_{k}}D_{R_{k}}V_{R_{k}}^{T}}_{R_{k}} = V_{R_{e,k}}D_{R_{e,k}}V_{R_{e,k}}^{T}.$$

When factors $V_{R_{e,k}}$ and $D_{R_{e,k}}$ are computed, the gain matrix can be found as follows:

$$K_{k} = \lambda_{k} P_{k|k-1} H_{k}^{T} R_{e,k}^{-1} = \lambda_{k} P_{k|k-1} H_{k}^{T} [V_{R_{e,k}}] [D_{R_{e,k}}^{1/2}]^{-2} [V_{R_{e,k}}]^{T}.$$

This validates the computation in line 3 of Algorithm 1d. Finally, the underlying formula for the $P_{k|k}$ calculation is the symmetric Joseph stabilized equation, i.e. it is the same as in Algorithm 1c. This completes the proof of Algorithm 1d.

As discussed in [26], such organization of computations (see Algorithm 1d) improves numerical stability of SVD-based filtering with respect to roundoff errors, because less matrix inversions are required. More precisely, only $D_{R_{e,k}}^{-1}$ is involved in Algorithm 1d, while $D_{P_{k|k-1}}^{-1}$ and $D_{P_{k|k}}^{-1}$ are not required.

Property	Cholesky-base	d methods	UD-based	methods	SVD-based methods			
	MCC-KF	IMCC-KF	MCC-KF IMCC-KF		MCC-KF		IMCC-KF	
	Alg. 1a	[19, Alg. 2]	Alg. 1b	Alg. 2b	Alg. 1c	Alg. 1d	Alg. 2c	
1. Type	Covariance	Covariance	Covariance	Covariance	Covariance	Covariance	Covariance	
2. Decomposition	Cholesky	Cholesky	modified Cholesky		SVD	SVD	SVD	
3. Restriction	$\Pi_0 > 0, Q_k >$	$0, R_k > 0$	$\Pi_0 > 0, Q_k > 0, R_k > 0$		no	no	no	
4. Pre-array	TU: any QR(1)	any QR(1)	MWGS(1)	MWGS(1)	SVD (1)	SVD (1)	SVD (1)	
factorization	MU: any QR(2)	any QR(1)	MWGS(2)	MWGS(1)	SVD (2)	SVD (2)	SVD (1)	
5. Matrix	$P_{k k-1}^{-1/2}, P_{k k}^{-1/2}$	$R_{e,k}^{-1/2}$	$ar{U}_{P_{k k-1}}^{-1},ar{U}_{P_{k k}}^{-1},$	$ar{U}_{R_{e,k}}^{-1}$	$D_{P_{k k-1}}^{-1/2}, D_{P_{k k}}^{-1/2}$	$D_{R_{e,k}}^{-1/2}$	$D_{P_{k k-1}}^{-1/2}, D_{P_{k k}}^{-1/2}$	
inversions			$D_{P_{k k-1}}^{-1}, D_{P_{k k}}^{-1}$					
6. Extended form	-	[19, Alg. 3]	-	?	-	_	?	

Table 1: Theoretical comparison of the factored-form implementations developed for the MCC-KF (Algorithm 1) and IMCC-KF (Algorithm 2) estimators.

Unfortunately, it is not possible to design the similar robust SVD-based variant for the IMCC-KF (Algorithm 2). More precisely, the goal is to avoid $D_{P_{kk-1}}^{-1}$ and $D_{P_{kk}}^{-1}$ operations in Algorithm 2c. Clearly, the gain computation K_k can be performed at the same way as it is done in Algorithm 1d, i.e. it requires $D_{R_{ek}}^{-1}$ computation, additionally. Having computed K_k , the SVD factors of the error covariance matrix $P_{k|k}$ should be updated through SVD factorization of the related pre-array, say A. For that, the underlying equation for $P_{k|k}$ should have a symmetric form, because any covariance matrix is symmetric, i.e. the related SVD is apllied to the symmetric pre-arrays product $A^{T}A$ or AA^{T} . For the IMCC-KF method, there are three possibilities for computing $P_{k|k}$, given by equations (14) – (16). Only formula (14) has the required symmetric form and it is already used in Algorithm 2c implying the calculation of $D_{P_{kk-1}}^{-1}$ and $D_{P_{kk}}^{-1}$. Equation (16) might be symmetric if one skips the scalar parameter λ_k . However, in this case the Joseph stabilized equation is obtained, i.e. we get the MCC-KF implementation (Algorithm 1 and its SVD-based variant in Algorithm 1d), but not the IMCC-KF method in Algorithm 2. The author still does not know how to balance equation (16), i.e. to express it in symmetric form that is appropriate for deriving the robust SVDbased implementation of the IMCC-KF (Algorithm 2). This is an open question for a future research.

5. Discussion and comparison

5.1. Theoretical comparison

Table 1 illustrates some theoretical aspects of the suggested factored-form filters' families. The following notation is used: sign "+" means that the corresponding property is available, sign "-" implies missing corresponding feature, and "?" means that the factored-form implementation with the related property might be derived in future.

Having analyzed the information presented in Table 1, we make the following conclusions. First, all filtering algorithms developed in this paper are of *covariance*-type. This means that the error covariance matrix $P_{k|k}$ (or its factors) are updated according to the underlying filter recursion. An alternative class of methods implies $\Lambda_{k|k} = P_{k|k}^{-1}$ propagation (called the information matrix) rather than $P_{k|k}$. Such algorithms are known as *information*-type implementations and they have some benefits over the covariance recursions. One of the main reason

to derive such implementations is a need to solve the state estimation problem without a *prior* information. In this case the initial error covariance matrix Π_0 is too "large" and, hence, the initialization step $\Pi_0 := \infty$ yields various complications for covariance filtering, while the information algorithms simply imply $\Lambda_0 := 0$. Additionally, the information filtering suggests a possible solution to numerical instability problem caused by influence of roundoff errors at the measurement update stage as discussed in [20, p. 356-357]. Further argument for deriving information filter recursion under the MCC approach is the matrix inversion $P_{k|k-1}^{-1}$ required at each iterate while computing the inflation parameter λ_k . To avoid this operation, it might be useful to derive the algebraic equivalent counterpart that updates the inverse (information) matrices (or their factors) automatically. It is worth noting here that similar motivation was used for developing information filtering for the classical KF in [39]. All these facts make the information-type implementations attractive for practical use. To the best of the author's knowledge, the information MCC KF-like methods still do not exist. Their derivation could be an area for a future research.

The second row in Table 1 summarizes the decomposition of covariance matrices involved in each implementation under examination. The type of factorization may impose restrictions on their properties. For instance, the Cholesky decomposition is known to exist and to be unique when the symmetric matrix to be decomposed is positive definite [40]. These conditions are presented in the third row of Table 1. In general, covariance is a positive semi-definite matrix and the Cholesky decomposition still exists for such matrices, however, it is not unique [41]. In this case, the Cholesky-based implementations are unexpectedly interrupted by the procedure performing the decomposition. From this point of view, the SVD-based filtering might be preferable because no restrictions are implied for performing SVD; see [38, Theorem 1.1.6]. Additionally, the SVD is the most accurate method to factorize the error covariance matrix (especially when it is close to singular), although it is more time consuming than the Cholesky decomposition.

Concerning the computational time, we conclude that the factored-form IMCC-KF implementations (Algorithms 2b, 2c and the previously published Algorithm 2 in [19]) are expected to work faster than the factored-form MCC-KF counterparts (Algorithms 1a, 1b, 1c, 1d), because they require less QR/MWGS/SVD factorizations at each filtering step. The pre-

cise number of computations required by each implementation depends on a particular QR, square-root-free QR (QRD algorithms, e.g. [31, 32]) and SVD methods utilized in practice. While deriving the factored-form implementations, no restriction on the pre-array transformations is imposed, i.e. the rotations can be implemented in various ways and, hence, the computational complexity analysis heavily depends on the users' choice and QR/MWGS/SVD method implemented.

Next, the required matrix inversions are outlined for each filtering algorithm in Table 1. As it has been already mentioned, it is preferable to avoid this operation in practice, for numerical and computational complexity reasons. The matrix R_k^{-1} required in calculating λ_k are not presented in Table 1, because this part is the same for all implementations, i.e. we take it out of the consideration. Having analyzed the information presented in Table 1, we conclude that the Cholesky- and UD-based IMCC-KF (see [19, Algorithm 2] and Algorithm 2b) are expected to possess a better numerical behavior than their MCC-KF counterparts (Algorithms 1a and 1b), because they require the inverse of $R_{e,k} \in \mathbb{R}^{m \times m}$ factors, only. Meanwhile Algorithms 1a and 1b involve the inverse of the error covariances $P_{k|k-1}$, $P_{k|k} \in \mathbb{R}^{n \times n}$ factors. Besides, if n >> m, then the Cholesky- and UD-based IMCC-KF algorithms are expected to be faster than the MCC-KF analogues (Algorithms 1a and 1b). However, for the SVD-based implementations this is not the case. Indeed, both Algorithm 1c and 2c are expected to be of the same robustness with respect to roundoff errors, because they imply the scalar divisions by square roots of the same singular values; see the terms $D_{P_{kk}}^{-1/2}$, $D_{P_{kk}}^{-1/2}$ in Algorithms 1c and 2c. In contrast, Algorithm 1d demands the inversion of diagonal matrix $D_{R_{ek}}^{1/2}$, only. Thus, it is expected to be more numerically stable with respect to roundoff errors than other SVD-based implementations, i.e. Algorithms 1c and 2c. In summary, only one SVD-based implementation has been found for the IMCC-KF estimator (Algorithm 2c). Meanwhile, two SVD-based implementations have been developed for the MCC-KF estimator (Algorithms 1c and 1d). Among these two methods, one implementation (Algorithm 1d) is expected to be the most numerically robust with respect to roundoff errors.

The final remark concerns the state vector $\hat{x}_{k|k}$ computation. We stress that the so-called *extended* array form is practically feasible for Cholesky-based IMCC-KF and it has been recently published in [19, Algorithm 3]. The key idea of such methods comes from the KF community where the extended array implementations exist for the Cholesky-based filtering [24] and for the UD-based methods [35, 36] while for the SVDbased algorithms this problem is still open [26]. The extended form implies an orthogonal transformation of *augmented* prearray [$\mathbb{A} \mid b$]. As a result, instead of explicit formula $\hat{x}_{k|k} =$ $\hat{x}_{k|k-1} + K_k(y_k - H_k \hat{x}_{k|k-1})$ for computing the state estimate, one utilizes a simple multiplication $\hat{x}_{k|k} = \left[P_{k|k}^{T/2}\right] \left[P_{k|k}^{-T/2} \hat{x}_{k|k}\right]$ of the blocks $\left[P_{k|k}^{1/2}\right]$ and $\left[P_{k|k}^{-T/2} \hat{x}_{k|k}\right]$ that are directly read-off from the corresponding *extended* post-array [$\mathbb{R} \mid \tilde{b}$]. This trick is intended to avoid any matrix inversion in the underlying filter recursion. In particular, in [19, Algorithm 2] the Cholesky factor $R_{e,k}^{-1/2}$ is

required for calculating $\hat{x}_{k|k}$, meanwhile the extended version in [19, Algorithm 3] does not involve it. Readers are referred to [19] for more details and proof. Here we would like to discuss the possibility to design such methods for other factoredform MCC KF-like estimators. Our first question is whether or not the extended array implementations are practically feasible for the original MCC-KF recursion (Algorithm 1). We answer negatively for this question, because as mentioned in Section 2, the equations for computing K_k and $P_{k|k}$ in Algorithm 1 are taken from two different sources: the error covariance $P_{k|k}$ is computed by the classical KF equation (Joseph stabilized form), meanwhile the filter gain K_k is calculated under the MCC methodology with implicated λ_k parameter. In summary, these formulas have difference nature and cannot be collected altogether into unique array that is a crucial point for derivation of extended array implementations. Thus, the sign "-" is mentioned in the last row of Table 1 for all factored-form MCC-KF variants. Meanwhile for the IMCC-KF (Algorithm 2) recursion the extended array algorithms seems to be possible to derive. The Cholesky-based method has been recently suggested in [19, Algorithm 3]. The question about existence of extended array UD- and SVD-based IMCC-KF implementations is still open. This can be an area for future research.

5.2. Numerical comparison

To justify the theoretical derivation of the suggested factoredform implementations, a linear stochastic state-space model for electrocardiogram signal processing [42] is explored. In contrast to the cited paper, the filtering methods are examined in the presence of impulsive noise/shot noise [11].

Example 1. The system state is defined as $x(t) = [s(t), \dot{s}(t), \ddot{s}(t)]^T$ where s(t) is the displacement of the object or signal at time t, the derivatives $\dot{s}(t)$ and $\ddot{s}(t)$ represent the velocity and acceleration, respectively. The discrete-time version of the model dynamic is given as follows:

$$x_k = \begin{bmatrix} 1 & \Delta t & \frac{(\Delta t)^2}{2} \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} x_{k-1} + w_{k-1}, \qquad x_0 \sim \mathcal{N}(\bar{x}_0, \Pi_0)$$

where $\Delta t = 0.1$ and $\bar{x}_0 = [1, 0.1, 0]^T$, $\Pi_0 = 0.1 I_3$. The dynamic is observed via the measurement scheme

$$y_k = \begin{vmatrix} 1 & 0 & 0 \end{vmatrix} x_k + v_k.$$

The entries of w_k and v_k are generated as follows:

 $w_k \sim \mathcal{N}(0, Q) + \text{Shot noise},$ $v_k \sim \mathcal{N}(0, R) + \text{Shot noise}$

where the covariances Q and R are

$$Q = \begin{bmatrix} \frac{1}{20}(\Delta t)^5 & \frac{1}{8}(\Delta t)^4 & \frac{1}{6}(\Delta t)^3\\ \frac{1}{8}(\Delta t)^4 & \frac{1}{3}(\Delta t)^3 & \frac{1}{2}(\Delta t)^2\\ \frac{1}{6}(\Delta t)^3 & \frac{1}{2}(\Delta t)^2 & \Delta t \end{bmatrix} \quad and \quad R = 0.01.$$

	MCC-KF	Factored-form family for MCC-KF				IMCC-KF	Factored-form family for IMCC-KF		
-	(conventional)	Cholesky-	UD-	SVD-	SVD-	(conventional)	Cholesky-	UD-	SVD-
	Algorithm 1	(1a)	(1b)	(1c)	(1d)	Algorithm 2	[19, Alg. 2]	(2b)	(2c)
$RMSE_{x_1}$	7.4691	7.4691	7.4691	7.4691	7.4691	7.3569	7.3569	7.3569	7.3569
$RMSE_{x_2}$	12.4615	12.4615	12.4615	12.4615	12.4615	12.3938	12.3938	12.3938	12.3938
$RMSE_{x_3}$	13.8206	13.8206	13.8206	13.8206	13.8206	13.7631	13.7631	13.7631	13.7631
$\ \mathbf{RMSE}_{x_i}\ _2$	20.0521	20.0521	20.0521	20.0521	20.0521	19.9287	19.9287	19.9287	19.9287
CPU (s)	0.0344	0.0543	0.0567	0.0680	0.0624	0.0264	0.0435	0.0472	0.0566

Table 2: The RMSE errors and average CPU time (s) for the MCC-KF and IMCC-KF implementations in Example 1, M = 500 Monte Carlo simulations.

To simulate the impulsive noise (shot noise), we follow the approach suggested in [11]. The Matlab routine Shot_noise recently published in [43, Appendix] can be used as follows: (i) only 10% of samples are corrupted by the outliers; (ii) the discrete time instants t_k corrupted by the outliers are selected randomly from the uniform discrete distribution in the interval [11, N - 1], i.e. the first ten and last time instants are not corrupted in our experiments; (iii) the outliers are all taken at different time instants; (iv) the magnitude of each impulse is chosen randomly from the uniform discrete distribution in the interval [0, 3]. Following [11], our routine additionally returns the sample covariances \hat{Q} and \hat{R} of the simulated random sequence. They are utilized by all estimators under examination.

To decide about estimation quality of each filtering method, the following numerical experiment is performed for 500 Monte Carlo runs: (1) the stochastic model is simulated for N = 300discrete-time points to generate the measurements, (2) the inverse problem (i.e. the estimation problem) is solved by various filtering methods with the same measurement history, the same initial conditions, the same adaptive kernel size selection approach published previously for the MCC-KF method in [11] and the same noises' covariances; (3) the root mean square error (RMSE) is calculated over 500 Monte Carlo runs as follows:

$$\text{RMSE}_{x_i} = \sqrt{\frac{1}{MN} \sum_{j=1}^{M} \sum_{k=1}^{N} \left(x_{i,exact}^j(t_k) - \hat{x}_{i,k|k}^j \right)^2}$$

where M = 500 is the number of Monte-Carlo trials, N = 300 is the discrete time of the dynamic system, the $x_{i,exact}^{j}(t_k)$ and $\hat{x}_{i,k|k}^{j}$ are the *i*-th entry of the "true" state vector (simulated) and its estimated value obtained in the *j*-th Monte Carlo run, respectively. The resulted $||\text{RMSE}_{x_i}||_2$ values are summarized in Table 2 for each implementation under assessment. The averaged CPU time (s) is also presented for each estimator.

Having analyzed the obtained results collected at the first panel, we conclude that all MCC-KF implementation methods derived in this paper are mathematically equivalent, i.e. the correctness of their derivation is substantiated by numerical experiments. The same conclusion holds for all IMCC-KF algorithms. Next, we observe that the SVD-based implementations are the most time consuming methods. As a benefit, we may mention that SVD provides an extra information about the matrix structure and, hence, these implementations might be used in various reduced-filters design strategies. The conventional algorithms are the most fast implementations, however, they are the most numerically unstable in ill-conditioned situations as we observe it in Example 2 below. Finally, having compared the results in the first and the second panels, we note that the IMCC-KF estimator (and all its factored-form implementations) outperforms the MCC-KF (and all its factored-form implementations, as well) for estimation accuracy. Indeed, the total RMSE of the IMCC-KF is less than the total RMSE of the MCC-KF method. The difference between them is caused by the neglected scaling parameter λ_k in equation (16), i.e. this is the price to be paid for keeping the symmetric Joseph stabilized formula for $P_{k|k}$ calculation in the MCC-KF estimator.

Unfortunately, Example 1 does not allow for exploring numerical insights of the examined implementations. To do so, a set of ill-conditioned test problems is considered in Example 2.

Example 2. The dynamic equation in Example 1 is observed via the following ill-conditioned scheme:

$$y_k = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 + \delta \end{bmatrix} x_k + v_k, \quad R \sim \mathcal{N}(0, \delta^2 I_2)$$

with the initial state $x_0 \sim \mathcal{N}(0, I_3)$ and in the presence of Gaussian uncertainties, only. Additionally, the ill-conditioning parameter δ is used for simulating roundoff and assumed to be $\delta^2 < \epsilon_{roundoff}$, but $\delta > \epsilon_{roundoff}$ where $\epsilon_{roundoff}$ denotes the unit roundoff error².

The set of numerical experiments described above for Example 1 is performed for Example 2 as well, except that the covariance matrix of measurement noise *R* remains the same, i.e. the process covariance *Q* is replaced by the sample covariance \hat{Q} , only. The resulted $\|\text{RMSE}_{x_i}\|_2$ values are summarized in Table 3 for each implementation under assessment and each value of parameter δ while it tends to machine precision limit.

Having analyzed the numerical results collected in Table 3, we conclude that all filters produce accurate estimates of the state vector while the estimation problem is well-conditioned, i.e. for large values of δ . The resulted accuracy of the MCC-KF and IMCC-KF techniques is quite similar, until the problem becomes ill-conditioned and all implementations start to diverge. Recall, the difference in the original MCC-KF and the IMCC-KF is in the equation for $P_{k|k}$, only.

It is important to compare the factored-form implementations within the MCC-KF and IMCC-KF techniques, separately. For

²Computer roundoff for floating-point arithmetic is often characterized by a single parameter $\epsilon_{roundoff}$, defined as the largest number such that either 1 + $\epsilon_{roundoff} = 1$ or 1 + $\epsilon_{roundoff}/2 = 1$ in machine precision.

Ill-conditioning	MCC-KF	Factored-form family for MCC-KF			IMCC-KF	Factored-form family for IMCC-KF			
parameter	(conventional)	Cholesky-	UD-	SVD-	SVD-	(conventional)	Cholesky-	UD-	SVD-
δ	Algorithm 1	(1a)	(1b)	(1c)	(1d)	Algorithm 2	[19, Alg. 2]	(2b)	(2c)
10^{-1}	0.1192	0.1192	0.1192	0.1192	0.1192	0.1209	0.1209	0.1209	0.1209
10^{-2}	0.1000	0.1000	0.1000	0.1000	0.1000	0.1023	0.1023	0.1023	0.1023
10^{-3}	0.1040	0.1040	0.1040	0.1040	0.1040	0.1066	0.1066	0.1066	0.1066
10^{-4}	0.1049	0.1049	0.1049	0.1049	0.1049	0.1069	0.1069	0.1069	0.1069
10^{-5}	0.1018	0.1018	0.1018	0.1018	0.1018	0.1047	0.1047	0.1047	0.1047
10^{-6}	0.0981	0.0981	0.0981	0.0981	0.0981	0.1008	0.1008	0.1008	0.1008
10^{-7}	0.0997	0.0996	0.0996	0.0997	0.0997	0.1025	0.1026	0.1026	0.1029
10 ⁻⁸	NaN	NaN	NaN	NaN	0.1016	NaN	0.1046	0.1046	72.7067
10 ⁻⁹	NaN	NaN	NaN	NaN	0.1004	NaN	0.1032	0.1032	Inf
10^{-10}	NaN	NaN	NaN	NaN	0.0915	NaN	0.0936	0.0937	NaN
10^{-11}	NaN	NaN	NaN	NaN	0.0868	NaN	0.0890	0.0788	NaN
10^{-12}	NaN	NaN	NaN	NaN	0.0997	NaN	0.1024	0.1021	NaN
10^{-13}	NaN	NaN	NaN	NaN	0.1003	NaN	0.1027	0.1026	NaN
10^{-14}	NaN	NaN	NaN	NaN	0.0985	NaN	0.1010	0.1009	NaN
10^{-15}	NaN	NaN	NaN	NaN	0.2341	NaN	0.1012	0.1011	NaN

Table 3: The effect of roundoff errors on the factored-form implementations designed for the MCC-KF (Algorithm 1) and IMCC-KF (Algorithm 2) estimators.

large δ , we observe that the *factored-form* algorithms produce absolutely the same results compared with their conventional counterparts in Algorithm 1 or 2, respectively. Again, this substantiates the algebraic equivalence between the suggested factored-form implementations and the corresponding conventional algorithms. While δ tends to machine precision limit, some numerical insights can be explored. More precisely, starting from $\delta = 10^{-7}$ and less, the factored-form implementations behave in different manner. The SVD-based Algorithms 1c and 2c suggested in this paper produce a slightly less accurate estimates than the Cholesky- and UD-based implementations until their divergence at $\delta = 10^{-8}$. This outcome was expected and discussed in details in previous section. Recall, both Algorithms 1c and 2c imply the scalar divisions by square roots of the same singular values; see the terms $D_{P_{klk-1}}^{-1/2}$ and $D_{P_{kk}}^{-1/2}$ involved. Thus, their numerical behaviour is similar. Meanwhile, among all suggested SVD-based implementations, Algorithms 1d is the most robust (with respect to roundoff errors) and it was anticipated in the previous section, as well. Indeed, the SVD-based MCC-KF implementation in Algorithm 1d maintains similar estimation accuracy as all other factored-form IMCC-KF implementations, i.e. the Choleskybased Algorithm 1a and the UD-based Algorithm 1b. We also conclude that the SVD-based implementation (Algorithm 1d) is the only one method in the MCC-KF factored-form family that manages the examined ill-conditioned situations in Example 2. In contrast, the Cholesky-based Algorithm 1a and UDbased Algorithm 1b are the most robust implementations in the factored-form family derived for the IMCC-KF estimator. Recall, the question whether or not it is possible to design similar robust SVD-based IMCC-KF variant is still open and to be investigated in future. As discussed in previous section, a non-symmetric form of equation (16) in the IMCC-KF estimator prevents the derivation.

Finally, we remark that all conventional implementations diverge at $\delta = 10^{-8}$. This conclusion holds for both the MCC-

KF (Algorithm 1) and IMCC-KF (Algorithm 2). The term NaN in Table 3 means that the estimator cannot solve the filtering problem since it produces no correct digits in the obtained state vector estimate. Furthermore, the obtained numerical results demonstrate divergence of all factored-form implementations of the MCC-KF (Algorithm 1), except the SVD-based variant in Algorithm 1d. Meanwhile, we observe an accurate solution produced by the Cholesky- and UD-based implementations of the IMCC-KF (Algorithm 2). In total, there are only three implementations that manage the ill-conditioned state estimation problem while $\delta \rightarrow \epsilon_{roundoff}$. In summary, the family of robust *factored-form* MCC-KF implementations (with respect to roundoff errors) consists of only SVD-based method in Algorithm 1d, while the robust IMCC-KF implementations are the Cholesky- and UD-based Algorithms 2a and 2b.

6. Concluding remarks

In this paper, complete families of the factored-form implementations are derived for both the MCC-KF and the IMCC-KF estimators. The theoretical discussion and the results of numerical experiments indicate that only Cholesky- and UD-based IMCC-KF implementations solve the ill-conditioned state estimation problem accurately. For the MCC-KF estimator, the robust SVD-based implementation exists and only this algorithm accurately treats the ill-conditioned cases.

A number of questions are still open for a future research. First, for the MCC-KF estimator, only one robust implementation was found in the factored-form family of reliable algorithms. This is the SVD-based implementation. Thus, the proposed Cholesky- and UD-based MCC-KF implementations are to be improved in future research, if possible. In contrast, for the IMCC-KF estimator, the robust Cholesky- and UD-based implementation are derived in the factored-form family. Meanwhile, the derivation of robust SVD-based implementation for the IMCC-KF estimator is still an open question. Recall, the problem is how to balance the equation for error covariance matrix calculation in order to derive a symmetric form that is similar to the Joseph stabilized equation proposed for the classical Kalman filter. Next, all algorithms derived in this paper are of covariance type. Meanwhile, the information filtering under the MCC approach still does not exist, i.e. neither the conventional recursion nor the factored-form implementations have been derived, yet. The adaptive kernel size selection strategy is another importance problem for all MCC KF-like filtering methods. The small kernel size might induce the instability problem, as well. Hence, the related stability issues (with respect to kernel size selection) should be investigated. Furthermore, the extended array implementations are of special interests for a future research, because of improved numerical stability caused by utilization of stable orthogonal rotations as far as possible. Finally, the derivation of stable factored-form implementations for solving nonlinear filtering problem under the maximum correntropy criterion via the accurate extended continuous-discrete KF approach presented recently in [45, 46, 47, 48, 49] is also planned for a future research.

Acknowledgments

The author thanks the support of Portuguese National Fund (*Fundação para a Ciência e a Tecnologia*) within the scope of project UID/Multi/04621/2013.

References

- A. Aravkin, J. V. Burke, L. Ljung, A. Lozano, G. Pillonetto, Generalized Kalman smoothing: Modeling and algorithms, Automatica 86 (2017) 63– 86.
- [2] C. Masreliez, R. Martin, Robust Bayesian estimation for the linear model and robustifying the Kalman filter, IEEE transactions on Automatic Control 22 (3) (1977) 361–371.
- [3] C. Hajiyev, H. E. Soken, Robust estimation of UAV dynamics in the presence of measurement faults, Journal of Aerospace Engineering 25 (1) (2010) 80–89.
- [4] L. Chang, B. Hu, G. Chang, A. Li, Robust derivative-free Kalman filter based on Huber's M-estimation methodology, Journal of Process Control 23 (10) (2013) 1555–1561.
- [5] B. A. Charandabi, H. J. Marquez, Observer design for discrete-time linear systems with unknown disturbances, in: IEEE 51st Annual Conference on Decision and Control, 2012, pp. 2563–2568.
- [6] B. A. Charandabi, H. J. Marquez, A novel approach to unknown input filter design for discrete-time linear systems, Automatica 50 (11) (2014) 2835–2839.
- [7] W. Liu, P. P. Pokharel, J. C. Príncipe, Correntropy: properties and applications in non-Gaussian signal processing, IEEE Transactions on Signal Processing 55 (11) (2007) 5286–5298.
- [8] G. T. Cinar, J. C. Príncipe, Hidden state estimation using the Correntropy filter with fixed point update and adaptive kernel size, in: The 2012 International Joint Conference on Neural Networks (IJCNN), 2012, pp. 1–6.
- [9] B. Chen, L. Xing, J. Liang, N. Zheng, J. C. Príncipe, Steady-state meansquare error analysis for adaptive filtering under the maximum correntropy criterion, IEEE Signal Processing Letters 21 (7) (2014) 880–884.
- [10] B. Chen, J. Wang, H. Zhao, N. Zheng, J. C. Príncipe, Convergence of a fixed-point algorithm under maximum correntropy criterion, IEEE Signal Processing Letters 22 (10) (2015) 1723–1727.
- [11] R. Izanloo, S. A. Fakoorian, H. S. Yazdi, D. Simon, Kalman filtering based on the maximum correntropy criterion in the presence of non-Gaussian noise, in: 2016 Annual Conference on Information Science and Systems (CISS), 2016, pp. 500–505.

- [12] B. Chen, X. Liu, H. Zhao, J. C. Príncipe, Maximum Correntropy Kalman Filter, Automatica 76 (2017) 70–77.
- [13] G. Y. Kulikov, M. V. Kulikova, Estimation of maneuvering target in the presence of non-Gaussian noise: A coordinated turn case study, Signal Processing 145 (2018) 241–257.
- [14] X. Liu, B. Chen, B. Xu, Z. Wu, P. Honeine, Maximum correntropy unscented filter, International Journal of Systems Science 48 (8) (2017) 1607–1615.
- [15] W. Qin, X. Wang, N. Cui, Maximum correntropy sparse Gauss-Hermite quadrature filter and its application in tracking ballistic missile, IET Radar, Sonar & Navigation 11 (9) (2017) 1388–1396.
- [16] D. Comminiello, J. C. Príncipe, Adaptive Learning Methods for Nonlinear System Modeling, 1st Edition, Elsevier, 2018.
- [17] B. Chen, J. C. Príncipe, Maximum correntropy estimation is a smoothed MAP estimation, IEEE Signal Processing Letters 19 (8) (2012) 491–494.
- [18] D. Simon, Optimal state estimation: Kalman, H-infinity, and nonlinear approaches, John Wiley & Sons, 2006.
- [19] M. V. Kulikova, Square-root algorithms for maximum correntropy estimation of linear discrete-time systems in presence of non-Gaussian noise, Systems & Control Letters 108 (2017) 8–15.
- [20] M. Grewal, A. Andrews, Kalman filtering: theory and practice using MATLAB, 4th Edition, John Wiley & Sons, New Jersey, 2015.
- [21] M. S. Grewal, J. Kain, Kalman filter implementation with improved numerical properties, IEEE Transactions on Automatic Control 55 (9) (2010) 2058–2068.
- [22] M. Morf, T. Kailath, Square-root algorithms for least-squares estimation, IEEE Trans. Automat. Contr. AC-20 (4) (1975) 487–497.
- [23] A. H. Sayed, T. Kailath, Extended Chandrasekhar recursion, IEEE Trans. Automat. Contr. AC-39 (3) (1994) 619–622.
- [24] P. Park, T. Kailath, New square-root algorithms for Kalman filtering, IEEE Trans. Automat. Contr. 40 (5) (1995) 895–899.
- [25] G. J. Bierman, Factorization Methods For Discrete Sequential Estimation, Academic Press, New York, 1977.
- [26] M. V. Kulikova, J. V. Tsyganova, Improved discrete-time Kalman filtering within singular value decomposition, IET Control Theory & Applications 11 (15) (2017) 2412–2418.
- [27] C. L. Thornton, Triangular covariance factorizations for Kalman filtering, Ph.D. thesis, University of California at Los Angeles (1976).
- [28] G. Bierman, C. Thornton, Numerical comparison of Kalman filter algorithms: Orbit determination case study, Automatica 13 (1) (1977) 23–35.
- [29] N. A. Carlson, Fast triangular formulation of the square root filter, AIAA journal 11 (9) (1973) 1259–1265.
- [30] A. Björck Solving least squares problems by orthogonalization, BIT 7 (1967) 1–21.
- [31] J. Götze, U. Schwiegelshohn, A square root and division free Givens rotation for solving least squares problems on systolic arrays, SIAM Journal on Scientific and Statistical Computing 12 (4) (1991) 800–807.
- [32] S. F. Hsieh, K. J. R. Liu, K. Yao, A unified square-root-free approach for QRD-based recursive-least-squares estimation, IEEE Transactions on Signal Processing 41 (3) (1993) 1405–1409.
- [33] A. Björck.
- [34] J. M. Jover, T. Kailath, A parallel architecture for kalman filter measurement update and parameter estimation, Automatica 22 (1) (1986) 43–57.
- [35] S. H. Chun, A single-chip QR decomposition processor for extended square-root Kalman filters, in: The 25-th Asilomar Conference on Signals, Systems and Computers, 1991, pp. 521–529.
- [36] I. V. Semushin, Y. V. Tsyganova, A. V. Tsyganov, E. F. Prokhorova, Numerically efficient UD filter based channel estimation for OFDM wireless communication technology, Procedia Engineering 201 (2017) 726–735.
- [37] L. Wang, G. Libert, P. Manneback, Kalman Filter Algorithm based on Singular Value Decomposition, in: Proceedings of the 31st Conference on Decision and Control, IEEE, Tuczon, AZ, USA, 1992, pp. 1224–1229.
- [38] A. Björck, Numerical methods in matrix computations, Springer, 2015.
- [39] P. Dyer, S. McReynolds, Extensions of square root filtering to include process noise, J. Opt. Theory Appl. 3 (6) (1969) 444–459.
- [40] G. H. Golub, C. F. Van Loan, Matrix computations, Johns Hopkins University Press, Baltimore, Maryland, 1983.
- [41] N. J. Higham, Analysis of the Cholesky decomposition of a semi-definite matrix, Tech. Rep. MIMS EPrint: 2008.56, University of Manchester (1990).
- [42] K. Suotsalo, S. Särkkä, A linear stochastic state space model for elec-

trocardiograms, in: The 27th IEEE International Workshop on Machine Learning for Signal Processing (MLSP), 2017.

- [43] M. V. Kulikova, Sequential maximum correntropy Kalman filtering, Asian Journal of Control 21 (6) (2019) 1–9.
- [44] H. E. Rauch, C. T. Striebel, F. Tung, Maximum likelihood estimates of linear dynamic systems, AIAA journal 3 (8) (1965) 1445–1450.
- [45] G. Yu. Kulikov, M. V. Kulikova, The accurate continuous-discrete extended Kalman filter for radar tracking, IEEE Trans. Signal Process. 64 (4) (2016) 948–958.
- [46] G. Yu. Kulikov, M. V. Kulikova, Estimating the state in stiff continuoustime stochastic systems within extended Kalman filtering, SIAM J. Sci. Comput. 38 (6) (2016) A3565–A3588.
- [47] G. Yu. Kulikov, M. V. Kulikova, Accurate cubature and extended Kalman filtering methods for estimating continuous-time nonlinear stochastic systems with discrete measurements, Appl. Numer. Math. 111 (2017) 260– 275.
- [48] G. Yu. Kulikov, M. V. Kulikova, Accurate continuous-discrete unscented Kalman filtering for estimation of nonlinear continuous-time stochastic models in radar tracking, Signal Process. 139 (2017) 25–35.
- [49] G. Yu. Kulikov, M. V. Kulikova, Moore-Penrose-pseudo-inverse-based Kalman-like filtering methods for estimation of stiff continuous-discrete stochastic systems with ill-conditioned measurementst, IET Control Theory Appl. 12 (16) (2018) 2205–2212.