Communications in Information and Systems
Vol. 5, No. 2, pp. 143-168, 2005

001

TWO-STAGE KALMAN FILTERING VIA STRUCTURED SQUARE-ROOT

STOYAN KANEV* AND MICHEL VERHAEGEN†

Abstract. This paper considers the problem of estimating an unknown input (bias) by means of the augmented-state Kalman (AKF) filter. To reduce the computational complexity of the AKF, [12] recently developed an optimal two-stage Kalman filter (TS-AKF) that separates the bias estimation from the state estimation, and shows that his new two-stage estimator is equivalent to the standard AKF, but requires less computations per iteration. This paper focuses on the derivation of the optimal two-stage estimator for the *square-root covariance implementation* of the Kalman filter (TS-SRCKF), which is known to be numerically more robust than the standard covariance implementation. The new TS-SRCKF also estimates the state and the bias separately while at the same time it remains equivalent to the standard augmented-state SRCKF. It is experimentally shown in the paper that the new TS-SRCKF may require less flops per iteration for some problems than the Hsieh's TS-AKF [12]. Furthermore a second, even faster (single-stage) algorithm has been derived in the paper by exploiting the structure of the least-squares problem and the square-root covariance formulation of the AKF. The computational complexities of the two proposed methods have been analyzed and compared the those of other existing implementations of the AKF.

Key words: two-stage augmented Kalman filter, square-root covariance implementation.

1. Introduction. This paper considers the problem of estimating an unknown, time-varying bias in the state equation of a time-variant stochastic discrete-time system in a minimum mean-square estimation error (MMSE) sense. A standard way of computing the MMSE estimate is by augmenting the state of the system with the unknown bias (after assuming a random walk model for the latter) and then estimating the resulting augmented state using a standard Kalman filter. This approach is called the augmented-state Kalman filter (AKF). A disadvantage of this approach is that the dimension of the covariance matrix increases so that the algorithm becomes computationally more involved and numerical inaccuracies can occur. To avoid this, a two-stage Kalman filter was proposed by Friedland in [5] that splits the state estimation from the unknown bias estimation under the assumption that the bias term is constant. While the approach of Friedland is very suitable for parallel computations, as argued by [12], if the bias is nonconstant the estimator becomes suboptimal in practice, i.e. it is not exactly equivalent to the AKF. To circumvent this problem, Hsieh and Chen [12] recently proposed a modification of the two-stage Kalman filter and proved its optimality also for stochastic/dynamic bias. In [18], a work that also appeared at that time, the authors develop an alternative optimal two-stage Kalman estimator under the weak assumption (which is not present in the work of Hsieh and

*Delft Center for Systems and Control, Mekelweg 2, 2628 CD Delft, The Netherlands. Tel. +31(0)15 27 86707, Fax. +31(0)15 27 86697, E-mail: {s.kanev; m.verhaegen}@dcsc.tudelft.nl

Chen) that the bias is uncorrelated with the process noise.

The idea behind the Hsieh's optimal two-stage Kalman filter (TS-AKF) is to apply a certain transformation on the augmented state and its covariance matrix in such a way that the transformed covariance matrix becomes block-diagonal. This property makes it possible to rewrite the augmented Kalman filter as two separate (but coupled!) filters, namely a bias-free estimator for the *transformed* system state and a bias estimator. It is shown in [12] that the TS-AKF exactly implements the AKF, and therefore achieves the MMSE estimate of the state and the bias. At the same time, the computational cost of new TS-AKF is usually much lower than that of the conventional AKF [12]. However, there exists a faster implementation of the AKF that was shown to be even less computationally demanding than the TS-AKF [20].

In this paper the same idea is used to derive a similar two-stage version of the *square-root covariance implementation* of the augmented Kalman filter, which is known to be numerically more reliable than than the standard covariance form [26]. The newly proposed algorithm, called here the two-stage square-root covariance Kalman filter (TS-SRCKF) consists similarly to the TS-AKF of two interacting square-root covariance Kalman filters. It is shown that the new algorithm is also equivalent to the AKF, but (in addition to the better numerical properties) can be computationally less involved than the Hsieh's TS-AKF for some problems. Furthermore, an even less computationally demanding algorithm is proposed in this paper by exploiting the underlying structure of the least-squares problem and the square-root covariance formulation of the AKF.

Since the work of Friedland [5] there has been a lot of interest in the state estimation problem for systems with unknown inputs. A number of alternative derivations of Friedland's two-stage Kalman filtered appeared in the late 80's [22, 2, 15, 25, 24, 23]. Friedland's estimator, however, is optimal only for constant bias. As shown by [1], for random bias the two-stage filter is optimal only when a certain algebraic constraint on the correlation between the state and the bias process is satisfied. A suboptimal solution in the case of stochastic bias was proposed in [16]. Friedland's two-stage bias filter has also been extended to nonlinear systems [21, 29]. However, as discussed above, it was only recently that an optimal two-stage Kalman estimator was developed for a general random/dynamical bias [12, 18, 17]. The optimal two-stage Kalman filter of Hsieh was subsequently extended in [13] to more general systems with no constraint on their structure, as well as to nonlinear systems by means of using the extended Kalman filter [11].

Another approach to state estimation in the presence of unknown bias is based on the design of filters that are decoupled from the bias [19, 9, 4, 10]. Reduced order estimators may also be used to further reduce the computational complexity [6, 8, 14]. Note also, that there is a clear link between the unknown input observers and the problem of detection and estimation of jump in the mean in systems. The goal there is to detect the time instant of occurrence of a jump in the state as well as to estimate its size. A very often used method for that purpose is the generalized likelihood ratio (GRL) approach [28, 7], where basically a nominal Kalman filter is used based on "no-jump" hypothesis from which the residual is used in a GLR scheme to detect and estimate the jump, if occurred. Subsequently, the state estimate from the Kalman filter is updated using the estimated jump. This paper, however, has only the purpose to discuss and compare different *implementations* of the augmented state Kalman filter which produce completely the same output (i.e. state-estimate and covariance matrix), where with the focus on their computational complexity. Hence, other alternatives as those mentioned above fall outside the scope of this paper.

2. Problem Formulation.

2.1. Notation. The following notation is used. $\mathbb{R}^n$ denotes the n-dimensional Euclidean space. The δ-function is defined as

$$\delta_{kt} \triangleq \begin{cases} 1, & k = t, \\ 0, & k \neq t. \end{cases}$$

The square-root of a symmetric positive (semi-)definite matrix P is defined as *any* matrix S such that $SS^T = P$. In order to avoid unnecessary definitions of new variables, the notation $P^{1/2}$ will also be used to denote a square root of the matrix P. Note that the square-root matrix S is not unique as the matrix SU is also a square-root of P for any unitary matrix U. The notation $\nu_k \frown (\bar{\nu}_k, Q_k^\nu)$ is used to denote a random Gaussian process ν_k with mean value $\bar{\nu}_k$ and covariance matrix Q_k^ν, which can also be written in the square-root covariance representation

$$\nu_k = \bar{\nu}_k + (Q_k^\nu)^{1/2} w_k, \quad w_k \frown (0, I),$$

where w_k is a zero-mean white Gaussian noise with covariance matrix equal to the identity matrix I. In matrices the symbol $\bigstar$ will be used to denote block entries of no importance for the discussion. The symbol $\bullet$ is used to denote matrices that can be implied by symmetry arguments. Finally, for a matrix S, the shorthand notation $S^2 = SS^T$ will also be used.

2.2. Problem Statement. In this paper we consider the problem of estimating the state x_k and the (random/dynamical) bias μ_k for the following discrete-time system

$$\begin{aligned} x_{k+1} &= A_k x_k + B_k \mu_k + \xi_k, \\ y_k &= C_k x_k + D_k \mu_k + \eta_k, \end{aligned} \tag{1}$$

where

(A_k, B_k, C_k, D_k)	are the (known) system matrices of appropriate dimensions,
$x_k \in \mathbb{R}^n$	is the system state,
$\mu_k \in \mathbb{R}^l$	unknown input (bias),
$y_k \in \mathbb{R}^p$	is the measured system output,
$\xi_k \frown (0, (Q_k^x)^2)$	is zero-mean process noise with covariance $E\{\xi_k \xi_t^T\} = (Q_k^x)^2 \delta_{kt}$,
$\eta_k \frown (0, (R_k)^2)$	is zero-mean measurement noise with covariance $E\{\eta_k \eta_t^T\} = (R_k)^2 \delta_{kt}$.

The starting point in the augmented state Kalman filter is the representation of the bias by means of a random walk model of the form

$$\mu_{k+1} = \mu_k + n_k, \tag{2}$$

with $n_k \frown (0, (Q_k^\mu)^2)$. As a result, equations (1) and (2) can be combined in the following augmented state model

$$\begin{aligned} X_{k+1} &= \bar{A}_k X_k + \bar{Q}_k \nu_k^x, \\ y_k &= \bar{C}_k X_k + R_k \nu_k^y, \end{aligned} \tag{3}$$

where $\nu_k^x \frown (0, I_{n+l})$, $\nu_k^y \frown (0, I_p)$, and

$$X_k \triangleq \begin{bmatrix} x_k \\ \mu_k \end{bmatrix}, \qquad \bar{A}_k \triangleq \begin{bmatrix} A_k & B_k \\ 0 & I \end{bmatrix},$$
$$\bar{C}_k \triangleq \begin{bmatrix} C_k & D_k \end{bmatrix}, \quad \bar{Q}_k \triangleq \begin{bmatrix} Q_k^x & Q^{x\mu_k} \\ 0 & Q_k^\mu \end{bmatrix}.$$

The matrix $\bar{Q}_k$ is assumed upper block-triangular without any loss of generality.

It is assumed in the paper that the pair $\{\bar{A}_k, \bar{C}_k\}$ is observable. A necessary condition for that is that the original pair $\{A_k, C_k\}$ is observable and that no eigenvalues of the matrix A lie on the unit circle.

The standard Kalman filter, used to estimate the state of the augmented system (3) is referred to as the *augmented-state Kalman Filter* (AKF). In this paper, however, instead of using the standard Kalman filter to estimate X_k we make use of a different, numerically more robust implementation called the *Square-Root Covariance Kalman Filter* (SRCKF). It is summarized in the next section.

2.3. The Square-Root Covariance Kalman Filter.

Algorithm 1 (SRCKF). *Given $\bar{A}_k$, $\bar{B}_k$, $\bar{C}_k$, $\bar{Q}_k$, R_k, $S_{k|k-1}$, $X_{k|k-1}$, and new measurement data y_k, compute:*

Measurement update:

1. Using the QR-decomposition find orthogonal matrix $\bar{T}_{RM}$ and matrices R_e and G_k such that

$$\begin{bmatrix} R_e & 0 \\ G_k & S_{k|k} \end{bmatrix} = \begin{bmatrix} \bar{C}_k S_{k|k-1} & -R_k \\ S_{k|k-1} & 0 \end{bmatrix} \bar{T}_{RM} \tag{4}$$

2. Form

$$X_{k|k} = X_{k|k-1} + G_k R_e^{-1}(y_k - \bar{C}_k \hat{X}_{k|k-1}). \tag{5}$$

Time update:

3. Using the QR-decomposition find orthogonal matrix $\bar{T}_{RT}$ and the square-root covariance matrix $S_{k+1|k}$ such that

$$\begin{bmatrix} S_{k+1|k} & 0 \end{bmatrix} = \begin{bmatrix} -\bar{A}_k S_{k|k} & \bar{Q}_k \end{bmatrix} \bar{T}_{RT} \tag{6}$$

4. Compute

$$X_{k+1|k} = \bar{A}_k X_{k|k}. \tag{7}$$

It has been shown [27] that the Kalman filter can equivalently be implemented as the solution of a least-squares problem making use of the square-root of the state covariance matrix. This SRCKF implementation is briefly explained here as it is the basis of the further developments in the paper. To begin with, assume that the state estimate at time instant k can be represented as

$$X_{k|k-1} = X_k + S_{k|k-1} w_k, \quad w_k \frown (0, I).$$

In other words, $X_k \frown (X_{k|k-1}, P_{k|k-1})$ with $P_{k|k-1} = S_{k|k-1} S_{k|k-1}^T$. Adding this equation to the augmented system (3) results in

$$\begin{bmatrix} X_{k|k-1} \\ y_k \\ 0_{(n+l)\times 1} \end{bmatrix} = \begin{bmatrix} I_{n+l} & 0_{n+l} \\ \bar{C}_k & 0_{p\times(n+l)} \\ \bar{A}_k & -I_{n+l} \end{bmatrix} \begin{bmatrix} X_k \\ X_{k+1} \end{bmatrix} + \begin{bmatrix} S_{k|k-1} & & \\ & R_k & \\ & & \bar{Q}_k \end{bmatrix} \nu_k,$$

with $\nu_k \frown (0, I_{2(n+l)+p})$. The SRCKF is based on the solution of the least-squares problem $\min_{X_k, X_{k+1}} \|\nu_k\|_2$ and is summarized in Algorithm 1. A very detailed treatment of the square-root covariance Kalman filter can be found in [27].

In the next section we show how the SRCKF can be split into two separate (but coupled) estimators.

3. Two-Stage Implementation of the SRCKF. In this section we show that the SRCKF, summarized in Algorithm 1, can be implemented as two separate

SRCKF's, one that estimates the bias and another that estimates the original system state (but in another basis). To begin with, we define the transformation matrix

$$T(M) \triangleq \begin{bmatrix} I & M \\ 0 & I \end{bmatrix},$$

that has the properties that $T(M_1)T(M_2) = T(M_1 + M_2)$ and therefore $T^{-1}(M) = T(-M)$. Then, given two matrices U_k and V_k (to be determined in what follows), we consider the following transformations

$$\bar{X}_{k|k-1} \triangleq T(-U_k)X_{k|k-1}, \tag{8}$$
$$\bar{X}_{k|k} \triangleq T(-V_k)X_{k|k}, \tag{9}$$
$$\bar{S}_{k|k-1} \triangleq T(-U_k)S_{k|k-1}, \tag{10}$$
$$\bar{S}_{k|k} \triangleq T(-V_k)S_{k|k}, \tag{11}$$

and the inverse transformations

$$X_{k|k-1} = T(U_k)\bar{X}_{k|k-1}, \tag{12}$$
$$X_{k|k} = T(V_k)\bar{X}_{k|k}, \tag{13}$$
$$S_{k|k-1} = T(U_k)\bar{S}_{k|k-1}, \tag{14}$$
$$S_{k|k} = T(V_k)\bar{S}_{k|k}. \tag{15}$$

The goal is to compute the matrices U_k and V_k in such a way that the transformed covariance matrices become block-diagonal, i.e.

$$\bar{P}_{k|k-1} \triangleq \bar{S}_{k|k-1}\bar{S}_{k|k-1}^T = \begin{bmatrix} \bar{P}_{k|k-1}^x & \\ & \bar{P}_{k|k-1}^\mu \end{bmatrix} \tag{16}$$
$$\bar{P}_{k|k} \triangleq \bar{S}_{k|k}\bar{S}_{k|k}^T = \begin{bmatrix} \bar{P}_{k|k}^x & \\ & \bar{P}_{k|k}^\mu \end{bmatrix}. \tag{17}$$

Before we proceed with finding expressions for the matrices U_k and V_k, we partition the transformed augmented state $\bar{X}$ as follows

$$\bar{X}_{k|k} \triangleq \begin{bmatrix} \bar{x}_{k|k} \\ \bar{\mu}_{k|k} \end{bmatrix}, \quad \bar{X}_{k|k-1} \triangleq \begin{bmatrix} \bar{x}_{k|k-1} \\ \bar{\mu}_{k|k-1} \end{bmatrix},$$

in conformance with the original augmented state. Using equations (6)-(5) we can write

$$
\begin{aligned}
\bar{X}_{k|k-1} &\overset{(8)}{=} T(-U_k)\bar{A}_{k-1}X_{k-1|k-1} \\
&\overset{(13)}{=} T(-U_k)\bar{A}_{k-1}T(V_{k-1})\bar{X}_{k-1|k-1}
\end{aligned} \tag{18}
$$

$$
\begin{aligned}
\bar{X}_{k|k} &\overset{(9)}{=} T(-V_k)(X_{k|k-1} + G_k R_e^{-1}(y_k - \bar{C}X_{k|k-1})) \\
&\overset{(12)}{=} T(U_k - V_k)\bar{X}_{k|k-1} \\
&\quad + T(-V_k)G_k R_e^{-1}(y_k - \bar{C}T(U_k)\bar{X}_{k|k-1}))
\end{aligned} \tag{19}
$$

$$
\begin{aligned}
\begin{bmatrix} \bar{S}_{k|k-1} & 0 \end{bmatrix} &\overset{(10)}{=} T(-U_k)\begin{bmatrix} -\bar{A}_{k-1}S_{k-1|k-1} & \bar{Q}_{k-1} \end{bmatrix}\bar{T}_{RT} \\
&\overset{(15)}{=} T(-U_k)\begin{bmatrix} -\bar{A}_{k-1}T(V_{k-1})\bar{S}_{k-1|k-1} & \bar{Q}_{k-1} \end{bmatrix}\bar{T}_{RT}
\end{aligned} \tag{20}
$$

$$
\begin{aligned}
\begin{bmatrix} R_e & 0 \\ T(-V_k)G_k & \bar{S}_{k|k} \end{bmatrix} &\overset{(11)}{=} \begin{bmatrix} \bar{C}_k S_{k|k-1} & -R_k \\ T(-V_k)S_{k|k-1} & 0 \end{bmatrix}\bar{T}_{RM} \\
&\overset{(14)}{=} \begin{bmatrix} \bar{C}_k T(U_k)\bar{S}_{k|k-1} & -R_k \\ T(U_k - V_k)\bar{S}_{k|k-1} & 0 \end{bmatrix}\bar{T}_{RM}.
\end{aligned} \tag{21}
$$

Now, let us first concentrate on equation (20). Since at iteration $(k-1)$ the square-root covariance $\bar{S}_{k-1|k-1}$ was made such that $\bar{P}_{k-1|k-1}$ is block diagonal, then one can easily show that there exists an orthogonal matrix $T_{k-1|k-1}$ such that

$$
\bar{S}_{k-1|k-1}T_{k-1|k-1} \triangleq \begin{bmatrix} \bar{S}^x_{k-1|k-1} & \\ & \bar{S}^\mu_{k-1|k-1} \end{bmatrix}. \tag{22}
$$

Therefore, equation (20) becomes equivalent to

$$
\begin{aligned}
\begin{bmatrix} \bar{S}_{k|k-1} & 0 \end{bmatrix} &= T(-U_k)\begin{bmatrix} -\bar{A}_{k-1}T(V_{k-1})\begin{bmatrix} \bar{S}^x_{k-1|k-1} & \\ & \bar{S}^\mu_{k-1|k-1} \end{bmatrix}, & \bar{Q}_{k-1} \end{bmatrix} \\
&\quad \cdot \begin{bmatrix} T^T_{k-1|k-1} & \\ & I \end{bmatrix}\bar{T}_{RT} \\
&= \begin{bmatrix} I & -U_k \\ 0 & I \end{bmatrix}\begin{bmatrix} -\begin{bmatrix} A_{k-1}\bar{S}^x_{k-1|k-1} & \bar{U}_k\bar{S}^\mu_{k-1|k-1} \\ 0 & \bar{S}^\mu_{k-1|k-1} \end{bmatrix}, & \begin{bmatrix} Q^x_{k-1} & Q^{x\mu}_{k-1} \\ & Q^\mu_{k-1} \end{bmatrix} \end{bmatrix} T_{RT},
\end{aligned} \tag{23}
$$

where the notation

$$
\bar{U}_k \triangleq A_{k-1}V_{k-1} + B_{k-1}, \tag{24}
$$

was introduced. In order to obtain $\bar{S}^\mu_{k|k-1}$ suppose that U_k is already selected such that $\bar{P}_{k|k-1}$ is block diagonal. There is nothing wrong with this assumption as the expression for $\bar{S}^\mu_{k|k-1}$ that we obtain is independent of U_k. Since $\bar{P}_{k|k-1}$ is block diagonal, similarly to (22) it can be shown that there exists an orthogonal matrix $T_{k|k-1}$ such that

$$
\bar{S}_{k|k-1}T_{k|k-1} \triangleq \begin{bmatrix} \bar{S}^x_{k|k-1} & \\ & \bar{S}^\mu_{k|k-1} \end{bmatrix}. \tag{25}
$$

Therefore, equation (23) can be rewritten as follows

$$
\begin{aligned}
(26)\qquad & \begin{bmatrix} \bar{S}^x_{k|k-1} & 0 & 0 \\ 0 & \bar{S}^\mu_{k|k-1} & 0 \end{bmatrix} \\
&= \begin{bmatrix} A_{k-1}\bar{S}^x_{k-1|k-1} & (\bar{U}_k - U_k)\bar{S}^\mu_{k-1|k-1} & Q^x_{k-1} & Q^{x\mu}_{k-1} - U_k Q^\mu_{k-1} \\ 0 & \bar{S}^\mu_{k-1|k-1} & 0 & Q^\mu_{k-1} \end{bmatrix} T^{(1)},
\end{aligned}
$$

where

$$
T^{(1)} = \begin{bmatrix} -T^T_{k-1|k-1} & \\ & I_{n+l} \end{bmatrix} \bar{T}_{RT} \begin{bmatrix} T_{k|k-1} & \\ & I_{n+l} \end{bmatrix}.
$$

Hence, the square-root covariance matrix $\bar{S}^\mu_{k|k-1}$ can be found by means of a QR-decomposition,

$$
(27)\qquad \boxed{\begin{bmatrix} \bar{S}^\mu_{k|k-1} & 0 \end{bmatrix} = \begin{bmatrix} \bar{S}^\mu_{k-1|k-1}, & Q^\mu_{k-1} \end{bmatrix} T^{(1)}_\mu.}
$$

To produce a similar expression for $\bar{S}^x_{k|k-1}$ we will first need to define U_k. In order to find the matrix U_k such that the matrix $\bar{P}_{k|k-1}$ becomes block-diagonal we first note, that

$$
\begin{aligned}
\bar{P}_{k|k-1} &= \bar{S}_{k|k-1}\bar{S}^T_{k|k-1} \\
&\overset{(20)}{=} T(-U_k)\bar{Q}^2_{k-1}T^T(-U_k) \\
&\quad + T(-U_k)\bar{A}_{k-1}T(V_{k-1})\bar{P}_{k-1|k-1}T^(V_{k-1})\bar{A}^T_{k-1}T^T(-U_k) \\
&= \begin{bmatrix} \star & (Q^{x\mu}_{k-1} - U_k Q^\mu_{k-1})(Q^\mu_{k-1})^T \\ \star & \star \end{bmatrix} \\
&\quad + \begin{bmatrix} I & -U_k \\ 0 & I \end{bmatrix} \begin{bmatrix} A_{k-1} & B_{k-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} I & V_{k-1} \\ 0 & I \end{bmatrix} \bar{P}_{k-1|k-1} \Big(\bullet\Big)^T \\
&\overset{(24)}{=} \begin{bmatrix} \star & (Q^{x\mu}_{k-1} - U_k Q^\mu_{k-1})(Q^\mu_{k-1})^T \\ \star & \star \end{bmatrix} \\
&\quad + \begin{bmatrix} A_{k-1} & \bar{U}_k - U_k \\ 0 & I \end{bmatrix} \bar{P}_{k-1|k-1} \Big(\bullet\Big)^T \\
&\overset{(17)}{=} \begin{bmatrix} \star & (Q^{x\mu}_{k-1} - U_k Q^\mu_{k-1})(Q^\mu_{k-1})^T \\ \star & \star \end{bmatrix} + \begin{bmatrix} \star & (\bar{U}_k - U_k)\bar{P}^\mu_{k-1|k-1} \\ \star & \star \end{bmatrix}.
\end{aligned}
$$

Therefore, in order that the upper off-diagonal term of $\bar{P}_{k|k-1}$ becomes equal to zero, U_k needs to be taken such that the following equality holds

$$
(\bar{U}_k - U_k)\bar{P}^\mu_{k-1|k-1} + (Q^{x\mu}_{k-1} - U_k Q^\mu_{k-1})(Q^\mu_{k-1})^T = 0
$$

that implies

$$
\begin{aligned}
U_k &= (\bar{U}_k\bar{P}^\mu_{k-1|k-1} + Q^{x\mu}_{k-1}(Q^\mu_{k-1})^T)(\bar{P}^\mu_{k-1|k-1} + (Q^\mu_{k-1})^2)^{-1} \\
&\overset{(27)}{=} (\bar{U}_k\bar{P}^\mu_{k|k-1} - \bar{U}_k(Q^\mu_{k-1})^2 + Q^{x\mu}_{k-1}(Q^\mu_{k-1})^T)(\bar{P}^\mu_{k|k-1})^{-1}.
\end{aligned}
$$

Therefore,

$$\boxed{U_k = \bar{U}_k + \big(Q^{x\mu}_{k-1}(Q^{\mu}_{k-1})^T - \bar{U}_k(Q^{\mu}_{k-1})^2\big)\big(\bar{S}^{\mu}_{k|k-1}(\bar{S}^{\mu}_{k|k-1})^T\big)^{-1}.} \tag{28}$$

Now that we have an expression for U_k we can proceed with finding an expression of $\bar{S}^x_{k|k-1}$ independent on $\bar{S}^{\mu}_{k|k-1}$. To this end, from equation (26) we write

$$\begin{aligned}
\bar{P}^x_{k|k-1} &= A_{k-1}\bar{P}^x_{k-1|k-1}A^T_{k-1} + (\bar{U}_k - U_k)\bar{P}^{\mu}_{k-1|k-1}(\bar{U}_k - U_k)^T + (Q^x_{k-1})^2 \\
&\quad + U_k(Q^{\mu}_{k-1})^2U^T_k + (Q^{x\mu}_{k-1})^2 - U_kQ^{\mu}_{k-1}(Q^{x\mu}_{k-1})^T - Q^{x\mu}_{k-1}(Q^{\mu}_{k-1})^TU^T_k \\
&= A_{k-1}\bar{P}^x_{k-1|k-1}A^T_{k-1} + (\bar{U}_k - U_k)(\bar{P}^{\mu}_{k-1|k-1} + (Q^{\mu}_{k-1})^2)(\bar{U}_k - U_k)^T \\
&\quad + (Q^x_{k-1})^2 + \bar{U}_k(Q^{\mu}_{k-1})^2U^T_k + U_k(Q^{\mu}_{k-1})^2\bar{U}^T_k - \bar{U}_k(Q^{\mu}_{k-1})^2\bar{U}^T_k + (Q^{x\mu}_{k-1})^2 \\
&\quad - U_kQ^{\mu}_{k-1}(Q^{x\mu}_{k-1})^T - Q^{x\mu}_{k-1}(Q^{\mu}_{k-1})^TU^T_k \\
&\overset{(27)}{=} A_{k-1}\bar{P}^x_{k-1|k-1}A^T_{k-1} + (\bar{U}_k - U_k)\bar{P}^{\mu}_{k|k-1}(\bar{U}_k - U_k)^T + (Q^x_{k-1})^2 \\
&\quad + \bar{U}_k(Q^{\mu}_{k-1})^2U^T_k + U_k(Q^{\mu}_{k-1})^2\bar{U}^T_k - \bar{U}_k(Q^{\mu}_{k-1})^2\bar{U}^T_k + (Q^{x\mu}_{k-1})^2 \\
&\quad - U_kQ^{\mu}_{k-1}(Q^{x\mu}_{k-1})^T - Q^{x\mu}_{k-1}(Q^{\mu}_{k-1})^TU^T_k.
\end{aligned} \tag{29}$$

Furthermore, from equation (28) we can write that

$$(\bar{U}_k - U_k)\bar{P}^{\mu}_{k|k-1} = \bar{U}_k(Q^{\mu}_{k-1})^2 - Q^{x\mu}_{k-1}(Q^{\mu}_{k-1})^T,$$

so that equation (29) becomes

$$\begin{aligned}
\bar{P}^x_{k|k-1} &= A_{k-1}\bar{P}^x_{k-1|k-1}A^T_{k-1} + \big(\bar{U}_k(Q^{\mu}_{k-1})^2 - Q^{x\mu}_{k-1}(Q^{\mu}_{k-1})^T\big)(\bar{U}_k - U_k)^T + (Q^x_{k-1})^2 \\
&\quad + \bar{U}_k(Q^{\mu}_{k-1})^2U^T_k + U_k(Q^{\mu}_{k-1})^2\bar{U}^T_k - \bar{U}_k(Q^{\mu}_{k-1})^2\bar{U}^T_k + (Q^{x\mu}_{k-1})^2 \\
&\quad - U_kQ^{\mu}_{k-1}(Q^{x\mu}_{k-1})^T - Q^{x\mu}_{k-1}(Q^{\mu}_{k-1})^TU^T_k \\
&= A_{k-1}\bar{P}^x_{k-1|k-1}A^T_{k-1} + (Q^x_{k-1})^2 + U_k(Q^{\mu}_{k-1})^2\bar{U}^T_k - Q^{x\mu}_{k-1}(Q^{\mu}_{k-1})^T\bar{U}^T_k \\
&\quad + (Q^{x\mu}_{k-1})^2 - U_kQ^{\mu}_{k-1}(Q^{x\mu}_{k-1})^T.
\end{aligned}$$

Therefore, the square-root covariance matrix can be obtained via the following QR-decomposition

$$\boxed{\begin{bmatrix}\bar{S}^x_{k|k-1} & 0\end{bmatrix} = \begin{bmatrix}A_{k-1}\bar{S}^x_{k-1|k-1}, & M_1^{1/2}\end{bmatrix}T^{(1)}_x,} \tag{30}$$

where it is denoted

$$M_1 = (Q^x_{k-1})^2 + (Q^{x\mu}_{k-1})^2 - Q^{x\mu}_{k-1}(Q^{\mu}_{k-1})^T\bar{U}^T_k + U_k\big(\bar{U}_k(Q^{\mu}_{k-1})^2 - Q^{x\mu}_{k-1}(Q^{\mu}_{k-1})^T\big)^T.$$

In order to find the matrix V_k we consider equation (21). By defining the matrix

$$S_k \triangleq C_kU_k + D_k, \tag{31}$$

equation (21) implies

$$
\begin{aligned}
&\begin{bmatrix} R_e R_e^T & R_e G_k^T T^T(-V_k) \\ T(-V_k) G_k R_e^T & \bar{P}_{k|k} + T(-V_k) G_k G_k^T T^T(-V_k) \end{bmatrix} \\
&= \begin{bmatrix} \begin{bmatrix} C_k & S_k \end{bmatrix} \bar{P}_{k|k-1} \begin{bmatrix} C_k & S_k \end{bmatrix}^T + R_k^2 & \begin{bmatrix} C_k & S_k \end{bmatrix} \bar{P}_{k|k-1} T^T(U_k - V_k) \\ T(U_k - V_k)\bar{P}_{k|k-1} \begin{bmatrix} C_k & S_k \end{bmatrix}^T & T(U_k - V_k)\bar{P}_{k|k-1} T^T(U_k - V_k) \end{bmatrix}.
\end{aligned} \tag{32}
$$

Therefore,

$$\bar{P}_{k|k} = T(U_k - V_k)\bar{P}_{k|k-1} T^T(U_k - V_k) - T(-V_k) G_k G_k^T T^T(-V_k).$$

The matrix V_k needs to be such that $\bar{P}_{k|k}$ becomes block-diagonal. Hence, considering block-entry (1,2) of $\bar{P}_{k|k}$, V_k should be such that the following equation holds

$$(U_k - V_k)\bar{P}^{\mu}_{k|k-1} - \begin{bmatrix} I & -V_k \end{bmatrix} G_k G_k^T \begin{bmatrix} 0 \\ I \end{bmatrix} = 0. \tag{33}$$

Let us introduce the notation

$$\begin{bmatrix} G_k^x \\ G_k^\mu \end{bmatrix} \triangleq T(-V_k) G_k = \begin{bmatrix} \begin{bmatrix} I & -V_k \end{bmatrix} G_k \\ \begin{bmatrix} 0 & I \end{bmatrix} G_k \end{bmatrix}. \tag{34}$$

Equation (33) then becomes

$$
\begin{aligned}
0 &= (U_k - V_k)\bar{P}^{\mu}_{k|k-1} - G_k^x R_e^{-1} R_e G_k^T \begin{bmatrix} 0 \\ I \end{bmatrix} \\
&\overset{(32)}{=} (U_k - V_k)\bar{P}^{\mu}_{k|k-1} - G_k^x R_e^{-1} \begin{bmatrix} C_k & S_k \end{bmatrix} \bar{P}_{k|k-1} T^T(U_k) \begin{bmatrix} 0 \\ I \end{bmatrix} \\
&= (U_k - V_k)\bar{P}^{\mu}_{k|k-1} - G_k^x R_e^{-1} \begin{bmatrix} C_k & S_k \end{bmatrix} \begin{bmatrix} 0 \\ \bar{P}^{\mu}_{k|k-1} \end{bmatrix} \\
&= (U_k - V_k)\bar{P}^{\mu}_{k|k-1} - G_k^x R_e^{-1} S_k \bar{P}^{\mu}_{k|k-1}.
\end{aligned}
$$

Therefore,

$$V_k = U_k - G_k^x R_e^{-1} S_k. \tag{35}$$

We will next obtain expressions for the square-root covariance matrices $\bar{S}^{\mu}_{k|k}$ and $\bar{S}^{x}_{k|k}$. To this end note that with the orthogonal matrix

$$T^{(2)} \triangleq \begin{bmatrix} T^T_{k|k-1} & \\ & -I_p \end{bmatrix} \bar{T}_{RM} \begin{bmatrix} I_p & \\ & T_{k|k} \end{bmatrix},$$

equation (21) can be expressed as follows

$$\begin{bmatrix} R_e & 0 & 0 \\ G_k^x & \bar{S}^x_{k|k} & 0 \\ G_k^\mu & 0 & \bar{S}^\mu_{k|k} \end{bmatrix} = \begin{bmatrix} C_k \bar{S}^x_{k|k-1} & S_k \bar{S}^\mu_{k|k-1} & R_k \\ \bar{S}^x_{k|k-1} & (U_k - V_k)\bar{S}^\mu_{k|k-1} & 0 \\ 0 & \bar{S}^\mu_{k|k-1} & 0 \end{bmatrix} T^{(2)}. \tag{36}$$

Therefore, the matrix $\bar{S}^{\mu}_{k|k}$ could be computed via the following QR-factorization

$$\boxed{\begin{bmatrix} R_e & 0 & 0 \\ G^{\mu}_k & \bar{S}^{\mu}_{k|k} & 0 \end{bmatrix} = \begin{bmatrix} S_k\bar{S}^{\mu}_{k|k-1} & C_k\bar{S}^{x}_{k|k-1} & R_k \\ \bar{S}^{\mu}_{k|k-1} & 0 & 0 \end{bmatrix} T^{(3)}.} \tag{37}$$

Thus, $\bar{S}^{\mu}_{k|k}$ depends also on $\bar{S}^{x}_{k|k-1}$. As for the matrix $\bar{S}^{x}_{k|k}$ an the expression can be obtained that does not depend on $\bar{S}^{\mu}_{k|k-1}$. Moreover this expression will further reduce the computational load for finding $\bar{S}^{x}_{k|k}$. To this end, we use equation (36) to write

(38)

$$\begin{bmatrix} R_eR_e^T & R_e(G^x_k)^T \\ G^x_kR_e^T & \bar{P}^x_{k|k} + G^x_k(G^x_k)^T \end{bmatrix}$$
$$= \begin{bmatrix} C_k\bar{P}^x_{k|k-1}C_k^T + S_k\bar{P}^{\mu}_{k|k-1}S_k^T + R_k^2 & \star \\ \bar{P}^x_{k|k-1}C_k^T + (U_k - V_k)\bar{P}^{\mu}_{k|k-1}S_k^T & \bar{P}^x_{k|k-1} + (U_k - V_k)\bar{P}^{\mu}_{k|k-1}(U_k - V_k)^T \end{bmatrix}.$$

Therefore,

$$\begin{aligned}
\bar{P}^x_{k|k} &= P^x_{k|k-1} + (U_k - V_k)\bar{P}^{\mu}_{k|k-1}(U_k - V_k)^T - G^x_k(G^x_k)^T \\
&= P^x_{k|k-1} + (U_k - V_k)\bar{P}^{\mu}_{k|k-1}(U_k - V_k)^T - G^x_kR_e^{-1}(G^x_kR_e^T)^T \\
&\overset{(38)}{=} P^x_{k|k-1} + (U_k - V_k)\bar{P}^{\mu}_{k|k-1}(U_k - V_k)^T \\
&\quad -G^x_kR_e^{-1}\big(\bar{P}^x_{k|k-1}C_k^T + (U_k - V_k)\bar{P}^{\mu}_{k|k-1}S_k^T\big)^T \\
&\overset{(35)}{=} P^x_{k|k-1} + G^x_kR_e^{-1}S_k\bar{P}^{\mu}_{k|k-1}(U_k - V_k)^T - G^x_kR_e^{-1}C_k\bar{P}^x_{k|k-1} \\
&\quad -G^x_kR_e^{-1}S_k\bar{P}^{\mu}_{k|k-1}(U_k - V_k)^T \\
&= P^x_{k|k-1} - G^x_kR_e^{-1}C_k\bar{P}^x_{k|k-1}. \qquad (39)
\end{aligned}$$

The last expression, however, still depends on $\bar{S}^{\mu}_{k|k-1}$ when the matrix $(G^x_kR_e^{-1})$ is computed using the QR-decomposition (36). However, $(G^x_kR_e^{-1})$ can also be obtained independent on $\bar{S}^{\mu}_{k|k-1}$ by means of a separate QR-decomposition. Indeed, note that from equation (38) we obtain

$$\begin{aligned}
G^x_kR_e^{-1} &= G^x_kR_e^T(R_eR_e^T)^{-1} \\
&= (\bar{P}^x_{k|k-1}C_k^T + (U_k - V_k)\bar{P}^{\mu}_{k|k-1}S_k^T) \\
&\quad \cdot(C_k\bar{P}^x_{k|k-1}C_k^T + S_k\bar{P}^{\mu}_{k|k-1}S_k^T + R_k^2)^{-1} \\
&\overset{(35)}{=} (\bar{P}^x_{k|k-1}C_k^T + G^x_kR_e^{-1}S_k\bar{P}^{\mu}_{k|k-1}S_k^T) \\
&\quad \cdot(C_k\bar{P}^x_{k|k-1}C_k^T + S_k\bar{P}^{\mu}_{k|k-1}S_k^T + R_k^2)^{-1}.
\end{aligned} \tag{40}$$

Therefore,

$$G^x_kR_e^{-1}(C_k\bar{P}^x_{k|k-1}C_k^T + S_k\bar{P}^{\mu}_{k|k-1}S_k^T + R_k^2) = \bar{P}^x_{k|k-1}C_k^T + G^x_kR_e^{-1}S_k\bar{P}^{\mu}_{k|k-1}S_k^T,$$

from where it follows that

$$G_k^x R_e^{-1} = \bar{P}^x_{k|k-1} C_k^T (C_k \bar{P}^x_{k|k-1} C_k^T + R_k^2)^{-1}.$$

In other words, if we define the matrices

$$\boxed{\begin{bmatrix} \bar{R}_e & 0 \\ \bar{G}_k^x & \bar{S}^x_{k|k} \end{bmatrix} = \begin{bmatrix} C_k \bar{S}^x_{k|k-1} & R_k \\ \bar{S}^x_{k|k-1} & 0 \end{bmatrix} T^{(4)},} \tag{41}$$

it can easily be seen that $\bar{G}_k^x \bar{R}_e^T = \bar{G}_k^x \bar{R}_e^{-1} \bar{R}_e \bar{R}_e^T = \bar{P}^x_{k|k-1} C_k^T$ and thus

$$\bar{G}_k^x (\bar{R}_e)^{-1} = G_k^x R_e^{-1} \tag{42}$$

and that equation (39) follows from (41). Note, that in view of (42) the expression for the matrix V_k in equation (35) can be rewritten as follows

$$\boxed{V_k = U_k - \bar{G}_k^x \bar{R}_e^{-1} S_k.} \tag{43}$$

In this way we have found suitable expressions for the square-root covariance matrices $\bar{S}^x_{k|k-1}$, $\bar{S}^\mu_{k|k-1}$, $\bar{S}^x_{k|k}$, $\bar{S}^\mu_{k|k}$, and the transformation matrices U_k and V_k. What remains is to obtain expressions for the state estimates $\bar{x}_{k|k-1}$, $\bar{\mu}_{k|k-1}$, $\bar{x}_{k|k}$, $\bar{\mu}_{k|k}$. We begin with the bias estimates:

$$\begin{aligned} \bar{\mu}_{k|k-1} &\overset{(18)}{=} \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} A_{k-1} & B_{k-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} I & V_{k-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} \bar{x}_{k-1|k-1} \\ \bar{\mu}_{k-1|k-1} \end{bmatrix} \\ &= \bar{\mu}_{k-1|k-1} \end{aligned} \tag{44}$$

$$\begin{aligned} \bar{\mu}_{k|k} &\overset{(19)}{=} \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} \bar{x}_{k|k-1} \\ \bar{\mu}_{k|k-1} \end{bmatrix} + \begin{bmatrix} 0 & I \end{bmatrix} G_k R_e^{-1} (y_k - \begin{bmatrix} C_k & S_k \end{bmatrix} \begin{bmatrix} \bar{x}_{k|k-1} \\ \bar{\mu}_{k|k-1} \end{bmatrix}) \\ &\overset{(34)}{=} \bar{\mu}_{k|k-1} + G_k^\mu R_e^{-1} (y_k - C_k \bar{x}_{k|k-1} - S_k \bar{\mu}_{k|k-1}). \end{aligned} \tag{45}$$

Similarly, the bias-free estimates can be obtained as follows

$$\begin{aligned} \bar{x}_{k|k-1} &\overset{(18)}{=} \begin{bmatrix} I & -U_k \end{bmatrix} \begin{bmatrix} A_{k-1} & B_{k-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} I & V_{k-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} \bar{x}_{k-1|k-1} \\ \bar{\mu}_{k-1|k-1} \end{bmatrix} \\ &= \begin{bmatrix} A_{k-1} & B_{k-1} - U_k \end{bmatrix} \begin{bmatrix} I & V_{k-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} \bar{x}_{k-1|k-1} \\ \bar{\mu}_{k-1|k-1} \end{bmatrix} \\ &= A_{k-1} \bar{x}_{k-1|k-1} + (\bar{U}_k - U_k) \bar{\mu}_{k-1|k-1}. \end{aligned} \tag{46}$$

$$\begin{aligned} \bar{x}_{k|k} &\overset{(19)}{=} \begin{bmatrix} I & U_k - V_k \end{bmatrix} \begin{bmatrix} \bar{x}_{k|k-1} \\ \bar{\mu}_{k|k-1} \end{bmatrix} + G_k R_e^{-1} (y_k - \begin{bmatrix} C_k & S_k \end{bmatrix} \begin{bmatrix} \bar{x}_{k|k-1} \\ \bar{\mu}_{k|k-1} \end{bmatrix}) \\ &\overset{(35)}{=} \bar{x}_{k|k-1} + G_k R_e^{-1} (y_k - C_k \bar{x}_{k|k-1}) \\ &\overset{(42)}{=} \bar{x}_{k|k-1} + \bar{G}_k \bar{R}_e^{-1} (y_k - C_k \bar{x}_{k|k-1}). \end{aligned} \tag{47}$$

Thus, we have derived two separate SRCKF's: a *bias filter* governed by equations (44), (45), (37), (27), and a *bias-free* filter given by equations (46), (47), (41), (30). These two filters are summarized in Algorithm 2 and Algorithm 3.

Algorithm 2 (Bias SRCKF)**.**

Input: *Given* C_k, D_k, Q_k^μ, R_k, $\bar{S}^\mu_{k|k-1}$, $\bar{S}^x_{k|k-1}$, U_k, $\bar{x}_{k|k-1}$, $\bar{\mu}_{k|k-1}$, *and new measurement data* y_k*:*

Define

$$S_k = C_k U_k + D_k. \tag{48}$$

Measurement update:

1. Using the QR-decomposition find orthogonal matrix $T^{(3)}$ *and matrices* R_e *and* G_k^μ *such*

$$\begin{bmatrix} R_e & 0 & 0 \\ G_k^\mu & \bar{S}^\mu_{k|k} & 0 \end{bmatrix} = \begin{bmatrix} S_k \bar{S}^\mu_{k|k-1} & C_k \bar{S}^x_{k|k-1} & R_k \\ \bar{S}^\mu_{k|k-1} & 0 & 0 \end{bmatrix} T^{(3)}. \tag{49}$$

2. Compute

$$\bar{\mu}_{k|k} = \bar{\mu}_{k|k-1} + G_k^\mu R_e^{-1}(y_k - C_k \bar{x}_{k|k-1} - S_k \bar{\mu}_{k|k-1}). \tag{50}$$

Time update:

3. Using the QR-decomposition find orthogonal matrix $T_\mu^{(1)}$ *and the square-root covariance matrix* $\bar{S}^\mu_{k+1|k}$ *such that*

$$\begin{bmatrix} \bar{S}^\mu_{k+1|k} & 0 \end{bmatrix} = \begin{bmatrix} \bar{S}^\mu_{k|k}, & Q_k^\mu \end{bmatrix} T_\mu^{(1)} \tag{51}$$

4. Compute

$$\bar{\mu}_{k+1|k} = \bar{\mu}_{k|k}. \tag{52}$$

Output: $\bar{\mu}_{k+1|k}$, $\bar{S}^\mu_{k+1|k}$, S_k.

Algorithm 3 (Bias-Free SRCKF).

Input: *Given* A_k, B_k, C_k, Q_k^x, $Q_k^{x\mu}$, Q_k^μ, R_k, $\bar{S}_{k+1|k}^\mu$, $\bar{S}_{k|k-1}^x$, S_k, U_k, $\bar{x}_{k|k-1}$, $\bar{\mu}_{k|k}$, *and new measurement data* y_k, *compute:*

Measurement update:

1. Using the QR-decomposition find orthogonal matrix $T^{(4)}$ *and matrices* $\bar{R}_e$ *and* $\bar{G}_k^\mu$ *such*

$$\begin{bmatrix} \bar{R}_e & 0 \\ \bar{G}_k^x & \bar{S}_{k|k}^x \end{bmatrix} = \begin{bmatrix} C_k \bar{S}_{k|k-1}^x & R_k \\ \bar{S}_{k|k-1}^x & 0 \end{bmatrix} T^{(4)}. \tag{53}$$

Define

$$V_k = U_k - \bar{G}_k^x \bar{R}_e^{-1} S_k \tag{54}$$

$$\bar{U}_{k+1} = A_k V_k + B_k \tag{55}$$

$$U_{k+1} = \bar{U}_{k+1} + (Q_k^{x\mu}(Q_k^\mu)^T - \bar{U}_{k+1}(Q_k^\mu)^2)(\bar{S}_{k+1|k}^\mu(\bar{S}_{k+1|k}^\mu)^T)^{-1} \tag{56}$$

2. Compute

$$\bar{x}_{k|k} = \bar{x}_{k|k-1} + \bar{G}_k^x \bar{R}_e^{-1}(y_k - C_k \bar{x}_{k|k-1}). \tag{57}$$

Time update:

3. Using the QR-decomposition find orthogonal matrix $T_x^{(1)}$ *and the square-root covariance matrix* $\bar{S}_{k+1|k}^x$ *such that*

$$\begin{aligned} M_1 = {} & (Q_k^x)^2 + (Q_k^{x\mu})^2 - Q_k^{x\mu}(Q_k^\mu)^T \bar{U}_{k+1}^T \\ & + U_{k+1}(\bar{U}_{k+1}(Q_k^\mu)^2 - Q_k^{x\mu}(Q_k^\mu)^T)^T \end{aligned} \tag{58}$$

$$\begin{bmatrix} \bar{S}_{k+1|k}^x & 0 \end{bmatrix} = \begin{bmatrix} A_k \bar{S}_{k|k}^x, & M_1^{1/2} \end{bmatrix} T_x^{(1)}. \tag{59}$$

4. Compute

$$\bar{x}_{k+1|k} = A_k \bar{x}_{k|k} + (\bar{U}_{k+1} - U_{k+1})\bar{\mu}_{k|k}. \tag{60}$$

Output: $\bar{x}_{k+1|k}$, $\bar{S}_{k+1|k}^x$, U_{k+1}.

Although not needed to run the two SRCKF's, the original (augmented) state-estimates and square-root covariance matrices can be restored by means of the inverse transformations (12), (13), (14), (15):

$$\begin{bmatrix} \hat{x}_{k|k} \\ \hat{\mu}_{k|k} \end{bmatrix} = \begin{bmatrix} \bar{x}_{k|k} + V_k \bar{\mu}_{k|k} \\ \bar{\mu}_{k|k} \end{bmatrix} \tag{61}$$

$$\begin{bmatrix} \hat{x}_{k+1|k} \\ \hat{\mu}_{k+1|k} \end{bmatrix} = \begin{bmatrix} \bar{x}_{k+1|k} + U_{k+1}\bar{\mu}_{k+1|k} \\ \bar{\mu}_{k+1|k} \end{bmatrix} \tag{62}$$

$$S_{k|k} = \begin{bmatrix} \bar{S}^x_{k|k} & V_k \bar{S}^\mu_{k|k} \\ 0 & \bar{S}^\mu_{k|k} \end{bmatrix} T^T_{k|k} \tag{63}$$

$$S_{k+1|k} = \begin{bmatrix} \bar{S}^x_{k+1|k} & U_{k+1} \bar{S}^\mu_{k+1|k} \\ 0 & \bar{S}^\mu_{k+1|k} \end{bmatrix} T^T_{k+1|k}. \tag{64}$$

Note that since the matrices $T_{k|k}$ and $T_{k+1|k}$ are unitary, the matrices $S_{k|k}T_{k|k}$ and $S_{k+1|k}T_{k+1|k}$ are also square-root covariance matrices.

4. Structured SRCKF. In this section we will discuss another implementation of the augmented-state SRCKF by making use of the structure (the sparseness of the matrices). This new algorithm, called here the Structured SRCKF, does not separate the bias estimation from the state estimation; it is based on the SRCKF applied to the whole augmented system. It will be shown later on, that this Structured SRCKF requires less flops per iteration that the TS-SRCKF, discussed above.

Consider the SRCKF, summarized in Algorithm 1. Clearly, most of the computational effort of the algorithm is concentrated in the two QR decompositions in equations (4) and (6). It has been shown in [27] that the square-root covariance implementation can equivalently rewritten in a combined time/measurement update

$$\begin{bmatrix} \bar{C}_k S_{k|k-1} & -R_k & 0 \\ \bar{A}_k S_{k|k-1} & 0 & -\bar{Q}_k \end{bmatrix} T_r = \begin{bmatrix} R^e_k & 0 & 0 \\ G_k & S_{k+1|k} & 0 \end{bmatrix}, \tag{65}$$

$$\hat{X}_{k+1|k} = \bar{A}_k \hat{X}_{k|k-1} + G_k (R^e_k)^{-1}(y_k - \bar{C}_k \hat{X}_{k|k-1}). \tag{66}$$

Since the matrix $S_{k|k-1}$ is lower triangular at each iteration, in order to keep the triangular structure also in the matrix $\bar{A}_k S_{k|k-1}$ we make $\bar{A}$ lower triangular by rewriting the augmented system (3) as follows

$$\begin{aligned} \begin{bmatrix} \mu_{k+1} \\ x_{k+1} \end{bmatrix} &= \begin{bmatrix} I & 0 \\ B_k & A_k \end{bmatrix} \begin{bmatrix} \mu_k \\ x_k \end{bmatrix} + \begin{bmatrix} Q^\mu_k & 0 \\ Q^{x\mu}_k & Q^x_k \end{bmatrix} \nu_k \\ y_k &= \begin{bmatrix} D_k & C_k \end{bmatrix} \begin{bmatrix} \mu_k \\ x_k \end{bmatrix} + R_k \xi_k. \end{aligned} \tag{67}$$

Assume, without loss of generality, that the matrices Q^μ_k and Q^x_k are lower triangular. Applying (65) to the new system (67) results in the matrix

$$M = \begin{pmatrix} \begin{bmatrix} D_k & C_k \end{bmatrix} \begin{bmatrix} S^\mu_{k|k-1} & 0 \\ S^{x\mu}_{k|k-1} & S^x_{k|k-1} \end{bmatrix} & -R_k & 0 \\ \begin{bmatrix} I & 0 \\ B_k & A_k \end{bmatrix} \begin{bmatrix} S^\mu_{k|k-1} & 0 \\ S^{x\mu}_{k|k-1} & S^x_{k|k-1} \end{bmatrix} & 0 & -\begin{bmatrix} Q^\mu_k & 0 \\ Q^{x\mu}_k & Q^x_k \end{bmatrix} \end{pmatrix}$$

that needs to be made lower triangular. To save computations we will now make use of the structure of the matrix M. We can write

$$M = \begin{bmatrix} D_k S^{\mu}_{k|k-1} + C_k S^{x\mu}_{k|k-1} & C_k S^{x}_{k|k-1} & -R_k & 0 & 0 & 0 \\ S^{\mu}_{k|k-1} & 0 & 0 & -Q^{\mu}_k & 0 & 0 \\ B_k S^{\mu}_{k|k-1} + A_k S^{x\mu}_{k|k-1} & A_k S^{x}_{k|k-1} & 0 & -Q^{x\mu}_k & -Q^{x}_k & 0 \end{bmatrix}.$$

Now, compute the following QR factorization

$$\begin{bmatrix} C_k S^{x}_{k|k-1} & -R_k & 0 \\ A_k S^{x}_{k|k-1} & 0 & -Q^{x}_k \end{bmatrix} T_1 = \begin{bmatrix} R^{\epsilon}_k & 0 & 0 \\ G^1_x & S^1_x & 0 \end{bmatrix}. \tag{68}$$

Then

$$M \sim \begin{bmatrix} D_k S^{\mu}_{k|k-1} + C_k S^{x\mu}_{k|k-1} & R^{\epsilon}_k & 0 & 0 \\ S^{\mu}_{k|k-1} & 0 & -Q^{\mu}_k & 0 \\ B_k S^{\mu}_{k|k-1} + A_k S^{x\mu}_{k|k-1} & G^1_x & -Q^{x\mu}_k & S^1_x \end{bmatrix}.$$

Next, compute the QR factorization making only the top two block rows lower triangular

$$\left[\begin{array}{ccc} D_k S^{\mu}_{k|k-1} + C_k S^{x\mu}_{k|k-1} & R^{\epsilon}_k & 0 \\ S^{\mu}_{k|k-1} & 0 & -Q^{\mu}_k \\ \hline B_k S^{\mu}_{k|k-1} + A_k S^{x\mu}_{k|k-1} & G^1_x & -Q^{x\mu}_k \end{array}\right] T_2 = \left[\begin{array}{ccc} R^{e}_k & 0 & 0 \\ G^{\mu}_k & S^{\mu}_{k+1|k} & 0 \\ \hline G^{x}_k & S^{x\mu}_{k+1|k} & U_3 \end{array}\right]. \tag{69}$$

Finally, compute

$$\begin{bmatrix} U_3 & S^1_x \end{bmatrix} T_3 = \begin{bmatrix} S^{x}_{k+1|k} & 0 \end{bmatrix}. \tag{70}$$

In this way we obtain

$$M \sim \begin{bmatrix} R^{e}_k & 0 & 0 & 0 \\ G^{\mu}_k & S^{\mu}_{k+1|k} & 0 & 0 \\ G^{x}_k & S^{x\mu}_{k+1|k} & S^{x}_{k+1|k} & 0 \end{bmatrix},$$

and then $\bar{X}_{k+1|k}$ can be computed from equation (66).

5. Computational Aspects. In this section we make a comparison between the number of flops (elementary additions and multiplications) per time instant for for the following implementations of the augmented-state Kalman filter:

- the conventional augmented-state Kalman filter (AKF), see e.g. [12].
- the structured augmented-state Kalman filter (Structured AKF), see [20], which is basically the conventional AKF where use is made of the structure of the matrix $\bar{A}_k$ (3),
- Hsieh's two-stage Kalman filter (TS-AKF) [12].

- the square-root covariance implementation of the augmented Kalman filter (SRCKF), as summarized in Algorithm 1.
- the square-root two-stage Kalman filter (TS-SRCKF), proposed in Algorithms 2-3.
- the structured square-root covariance Kalman filter (Structured SRCKF), proposed in Section 4.

In computing the number of flops we use the following algorithms:

QR decomposition The Householder algorithm [3, p.59] is used for computing a QR decomposition (see Algorithm 4).

Algorithm 4 (Householder QR decomposition).

For a matrix $A \in \mathbb{R}^{m\times n}$ do

```
for k=1:min(m,n)-1
     [uk,gk,shk] = house(A(k,k:n)');
     A(k,k:n) = [shk, zeros(1,n-k)];
     for j=k+1:m
          bjk=uk'*A(j,k:n)'/gk;
          A(j,k:n) = (A(j,k:n)' - bjk*uk)';
     end;
end;
RETURN A.
where the function HOUSE is defined as
function [u,g,sh] = house(a);
     s=sqrt(a'*a);
     sh=-sign(a(1))*s;
     u=a; u(1)=a(1)-sh;
     g=s*(s+abs(a(1)));
```

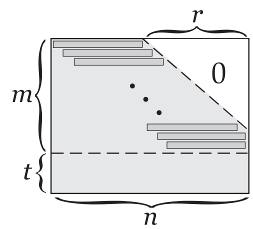
Using the Householder algorithm for performing a QR decomposition to make the first m rows of a matrix $A \in \mathbb{R}^{(m+t)\times n}$ $(m < n)$ lower-triangular, requires

$$\begin{aligned} f_{QR}(m,t,n) &= \sum_{k=1}^{m}\left(2(n-k+3)+\sum_{i=k+1}^{m+t}4(n-k+1)\right) \\ &= m(2n-m+5) \\ &\quad +2m\left(-\tfrac{1}{3}m^2+(n-t+1)m+(2n+1)t-n-\tfrac{2}{3}\right) \end{aligned}$$

flops.

The most significant reduction of the computational effort of the new algorithms is achieved by means of making use of the sparseness of the matrices during the QR decomposition.

For a (partially) trapezoidal matrix $M \in \mathbb{R}^{(m+t)\times n}$ with $m < n$ with the last r diagonals above the main diagonal equal to zero one can exploit the structure and compute the QR factorization in less flops, namely



$$\begin{aligned} f_{SQR}(m,t,n,r) &= f_{QR}(m,t,n) - f_{QR}(r,t,n) \\ &\quad +2r(n-r)(2m+2t-r) \\ &\quad +4r. \end{aligned}$$

Algorithm 5 is a modification of Algorithm 4 and computes the QR decomposition utilizing the trapezoidal structure of the matrix.

Algorithm 5 (Sparse Householder QR decomposition)**.**
For a partially trapezoidal matrix $A \in \mathbb{R}^{(m+t)\times n}$ with $(m < n)$ with zeros above the $(n-r)$-th diagonal do

```
for k=1:m
     [uk,gk,shk] = house(A(k,k:min(n,k+L))');
     A(k,k:n) = [shk, zeros(1,n-k)];
     for j=k+1:m+t
          aj=A(j,k:min(n,k+L))';
          bjk=uk'*aj/gk;
          A(j,k:min(n,k+L)) = (aj - bjk*uk)';
     end;
end;
RETURN A.
```

Inverse For finding the inverse of a square matrix $A \in \mathbb{R}^{n\times n}$ an algorithm based on Gauss elimination and backward substitution is used. This algorithm looks for a matrix M such that $AM = I$. For the i-th column of M we have $Am_i = e_i$ where $e_i = [0_{1\times(i-1)}, 1, 0_{1\times(n-i)}]^T$. Applying Gaussian elimination to the matrix $[A\ e_i]$ one gets $[\tilde{A},\ \tilde{e}_i]$ where $\tilde{A}$ is an upper-triangular matrix. Using subsequently backward substitution one obtains the elements of the vector m_i, which is such that $Am_i = e_i$ holds. The same applies for all columns of M. Obviously, it is not necessary to perform the same Gaussian elimination on A for all vectors m_i, one just needs to do this one time and apply the transformation on the elements of the identity matrix (see Algorithm 6).

Algorithm 6 (Gaussian Elimination).

For a nonsingular matrix $A \in \mathbb{R}^{n\times n}$ and matrix $B \in \mathbb{R}^{n\times n}$ do

```
for k=1:n-1
    for i=k+1:n
        lik=A(i,k)/A(k,k);
        for j=k+1:n
            A(i,k)=0;
            A(i,j)=A(i,j)-lik*A(k,j);
        end;
        B(i,:)=B(i,:)-lik*B(k,:);
    end;
end;
RETURN A and B.
```

Algorithm 7 (Backward Substitution).

For a nonsingular, upper-triangular matrix $A \in \mathbb{R}^{n\times n}$ and matrix $B \in \mathbb{R}^{n\times n}$ do

```
for k=n:-1:1
    Ainv(k,:) = B(k,:);
    for j=k+1:n
        Ainv(k,:) = Ainv(k,:) - A(k,j)*Ainv(j,:);
    end;
    Ainv(k,:) = Ainv(k,:) / A(k,k);
end;
RETURN Ainv.
```

The total number of flops (f_{GE}) required for performing Gaussian Elimination is equal to

$$\begin{aligned} f_{GE}(n) &= \sum_{k=1}^{n-1}\sum_{i=1}^{n}\left[1+\left(\sum_{j=k+1}^{n}2\right)+2n\right] \\ &= \frac{1}{6}n(n-1)(10n+1). \end{aligned}$$

The backward substitution (Algorithm 7), on its turn, requires additionally

$$f_{BS}(n) = \sum_{k=1}^{n} 2n(n-k)+n = n^3$$

flops. Therefore, the inverse of a full n-by-n matrix A costs

$$f_{INV}(n) = \frac{1}{6}n(16n^2+9n-1)$$

flops.

Since in the square-root covariance implementation of the Kalman filter we need only to invert triangular matrices, the first step in this algorithm (the Gaussian elimination) is not needed. To invert a triangular matrix only backward substitution is applied.

Square root The square root S of a matrix $P \in \mathbb{R}^{n\times n}$ is obtained using the Cholesky factorization [3, p.46], see Algorithm 8 that requires

$$\begin{aligned} f_{Ch}(n) &= \sum_{i=1}^{n}\left(2(i-1)+1+\sum_{k=1}^{n-i}2(i-1)+1\right)\\ &= \frac{1}{6}n(n+1)(2n+1) \end{aligned}$$

flops.

Algorithm 8 (Cholesky Factorization)**.**

For a nonsingular matrix $A \in \mathbb{R}^{n\times n}$ *do*

```
R=0;
for i=1:n,
    R(i,i) = 1/(A(i,i)-R(1:i-1,i)'*R(1:i-1,i));
    for j=i+1:n
    R(i,j) = (A(i,j)-R(1:i-1,i)'*R(1:i-1,j))/R(i,i);
    end;
end;
RETURN R.
```

Multiplication The triangular structure of $\bar{S}^{\mu}_{k+1|k}$ is utilized when computing U_{k+1}, namely that first the inverse, say S_{INV} of $\bar{S}^{\mu}_{k+1|k}$ is computed (which is also a triangular matrix) and next the product $S_{INV}S^T_{INV}$ is formed. Due to the triangular structure of S_{INV} this product can be computed in

$$f_{SS'}(l) = \sum_{i=1}^{l}\sum_{j=i}^{l}(2i-1) = \frac{1}{6}l(2l+1)(l+1)$$

flops. Similarly, the product between a full k-by-p matrix and a lower (upper) diagonal p-by-p matrix can be computed in

$$f_{GR_e}(k,p) = \sum_{j=1}^{p}2k(p-j) = 2kp^2 - kp(p+1)$$

flops. On the other hand, the multiplication of two full matrices $A \in \mathbb{R}^{n\times p}$ and $B \in \mathbb{R}^{p\times m}$ requires

$$M(n,p,m) = nm(2p-1)$$

flops.

Addition The flops required for adding two n-by-p matrices is obviously

$$A(n,p) = np.$$

Tables 1, 2 and 3 summarize the number of flops needed by the SRCKF, TS-SRCKF and Structured SRCKF algorithm, respectively, for performing one complete iteration, respectively. The number of flops required by the the other three compared algorithms (i.e. AKF, TS-AKF, and Structured AKF), have also been calculated, although not summarized here to avoid unnecessary details. Only the total number of flops, the number of additions and multiplications together, are compared.

TABLE 1
Number of flops needed by the SRCKF for completing one iteration.

operation	**equation**	**flops**
Form $\bar{C}_k S_{k\|k-1}$	-	$M(p, n+l, n+l)$
QR factorization	(4)	$f_{QR}(n+l+p, 0, n+l+p)$
R_e^{-1}	-	$f_{BS}(p)$
$G_k R_e^{-1}$	-	$M(n+l, p, p)$
innovation	-	$A(p,1) + M(p, n+l, 1)$
$A_k S_{k\|k}$	-	$M(n+l, n+l, n+l)$
QR factorization	(6)	$f_{QR}(n+l, 0, 2(n+l))$
$X_{k\|k}$	(5)	$A(n+l,1) + M(n+l, p, 1)$
$X_{k+1\|k}$	(7)	$M(n+l, n+l, 1)$

Table 4 summarizes the number of fops per time instant for the six compared Kalman filter implementations, computed for two different system dimensions (n, p, l). The percentages given between brackets represent the relative improvement with respect to the conventional AKF. Figure 1 depicts this relative reduction as a function of l and p for a fixed system order $n = 15$. Clearly, the algorithm that requires the least number of flops is the Structured SRCKF method, followed by the Structured AKF. The slowest algorithm, which is actually computationally more involved than the conventional AKF itself, is the SRCKF. The two-stage implementations, TS-AKF and TS-SRCKF, lie in between having approximately the same computational complexity (the TS-SRCKF needing less flops than TS-AKF only when $l > p$, i.e. when the dimension of the bias vector is larger than that of the number of measurements). It therefore become clear, that by simply exploiting the structure of the AKF and its square-root implementation (i.e. using the Structured SRCKF and Structured AKF) one can save much more computations than using the two-stage implementations (TS-SRCKF and TS-AKF).

Finally, we would like to discuss on the discussions that can be found in some other works. In particular, the results presented in this section confirm those previously

TABLE 2

Number of flops needed by the TS-SRCKF for completing one iteration.

operation	**equation**	**flops**
$(Q_k^x)^2, Q_k^{x\mu}(Q_k^\mu)^T$ $(Q_k^\mu)^2, (Q_k^{x\mu})^2$	-	$M(n,n,n)+M(l,l,l)+M(n,l,n+l)$
S_k	(48)	$M(p,n,l)+A(p,l)$
$\{C_k\bar{S}^x_{k\|k-1}, S_k\bar{S}^\mu_{k\|k-1}\}$	-	$f_{GR_e}(p,n)+f_{GR_e}(p,l)$
str. QR decomposition	(49)	$f_{SQR}(p+l,0,p+l+n,p-1)$
str. QR decomposition	(53)	$f_{SQR}(n+p,0,n+p,p-1)$
$\{R_e^{-1}, \bar{R}_e^{-1}\}$,	-	$2f_{BS}(p)$
$\{G_k^\mu R_e^{-1}, \bar{G}_k^x\bar{R}_e^{-1}\}$,	-	$f_{GR_e}(l,p)+f_{GR_e}(n,p)$
V_k	(54)	$A(n,l)+M(n,p,l)$
$\bar{U}_{k+1}$	(55)	$A(n,l)+M(n,n,l)$
$(\bar{S}^\mu_{k+1\|k})^{-1}$	-	$f_{BS}(l)$
$(\bar{S}^\mu_{k+1\|k})^{-1}(\bar{S}^\mu_{k+1\|k})^{-T}$	-	$f_{SS'}(l)$
U_{k+1}	(56)	$2A(n,l)+2M(n,l,l)$
$e_k^x = y_k - C_k\bar{x}_{k\|k-1}$	-	$A(p,1)+M(p,n,1)$
$e_k^\mu = e_k^x - S_k\bar{\mu}_{k\|k-1}$	-	$A(p,1)+M(p,l,1)$
$\bar{x}_{k\|k}$	(57)	$A(n,1)+M(n,p,1)$
$\bar{\mu}_{k\|k}$	(50)	$A(l,1)+M(l,p,1)$
$\{\bar{x}_{k+1\|k}, \bar{\mu}_{k+1\|k}\}$	(60),(52)	$M(n,n,1)+2A(n,l)+M(n,l,1)$
$A_k\bar{S}^x_{k\|k}$	-	$f_{GR_e}(n,n)$
M_1	(58)	$3A(n,n)+2M(n,l,n)$
$M_1^{1/2}$	-	$f_{Ch}(n)$
QR decomposition	(59)	$f_{QR}(n,0,2n)$
QR decomposition	(51)	$f_{QR}(l,0,2l)$
Computation of the original state and SRC		
$\{\hat{x}_{k\|k}, \hat{\mu}_{k\|k}\}$	(61)	$A(n,1)+M(n,l,1)$
$\{\hat{x}_{k+1\|k}, \hat{\mu}_{k+1\|k}\}$	(62)	$A(n,1)+M(n,l,1)$
$S^{12}_{k\|k}$	(63)	$f_{GR_e}(n,l)$
$S^{12}_{k+1\|k}$	(64)	$f_{GR_e}(n,l)$

summarized in [12] and [20], although both papers contain some small errors in the expressions of the number of flops. Still, similarly to what is stated in [12], the example here shows that the TS-AKF requires less flops than the *conventional* AKF, while the *Structured* AKF is even faster than the TS-AKF, as argued in [20].

6. Parallel Implementation of the TS-SRCKF. Apart from the reduced computational demand required by the TS-SRCKF, it is also very suitable for parallel

TABLE 3
Number of flops needed by the Structured SRCKF for completing one iteration.

operation	equation	flops
$D_k S^{\mu}_{k\|k-1}$	-	$f_{GR_e}(p,l)$
$C_k \begin{bmatrix} S^{x\mu}_{k\|k-1} & S^{x}_{k\|k-1} \end{bmatrix}$	-	$M(p,n,l) + f_{GR_e}(p,n)$
$B_k S^{\mu}_{k\|k-1}$	-	$f_{GR_e}(n,l)$
$A_k \begin{bmatrix} S^{x\mu}_{k\|k-1} & S^{x}_{k\|k-1} \end{bmatrix}$	-	$M(n,n,l) + f_{GR_e}(n,n)$
$\{A_k \hat{x}_{k\|k-1}, B_k \hat{\mu}_{k\|k-1}\}$	-	$M(n,n,1) + M(n,l,1)$
R_e^{-1}	-	$f_{BS}(p)$
$G_k R_e^{-1}$	-	$f_{GR_e}(n+l,p)$
innovation	-	$A(p,1) + M(p,n+l,1)$
$\hat{X}_{k+1\|k}$	(66)	$A(n+l,1) + M(n+l,p,1)$
str. QR factorization	(68)	$f_{SQR}(p+n,0,p+2n,p+n-1)$
str. QR factorization	(69)	$f_{SQR}(p+l,n,p+2l,l+p-1)$
str. QR factorization	(70)	$f_{SQR}(n,0,n+l,n-1)$

TABLE 4
Number of flops needed by the compared methods for completing one iteration.

Algorithm	**flops per iteration**	
	$n=5,\ p=5,\ l=5$	$n=15,\ p=5,\ l=10$
Covariance implementation		
AKF	10,845	119,820
Structured AKF	7,010 (↓ 35.4%)	66,320 (↓ 44.7%)
TS-AKF	10,230 (↓ 5.7%)	101,420 (↓ 15.4%)
Square-Root Covariance implementation		
SRCKF	12,065 (↑ 11.2%)	129,535 (↑ 8.1%)
TS-SRCKF	10,387 (↓ 4.2%)	93,347 (↓ 22.1%)
Structured SRCKF	6,393 (↓ 41.1%)	50,493 (↓ 57.9%)

implementation due to its decoupled structure. Table 5 illustrates how the TS-SRCKF can be implemented on two processors, where only at two instances data needs to be transferred from the processor implementing the bias-filter (left) to the processor that implements the bias-free filter (right). These two instances are the computation of the matrix U_{k+1} in equation (56), where the matrix $\bar{S}^{\mu}_{k+1|k}$ is needed, and the computation of $\bar{x}_{k+1|k}$ in (60), where $\bar{\mu}_{k|k}$ is necessary.

7. Conclusions. In this paper a two-stage implementation is developed for the augmented-state square-root covariance Kalman filter (TS-SRCKF) in the spirit of

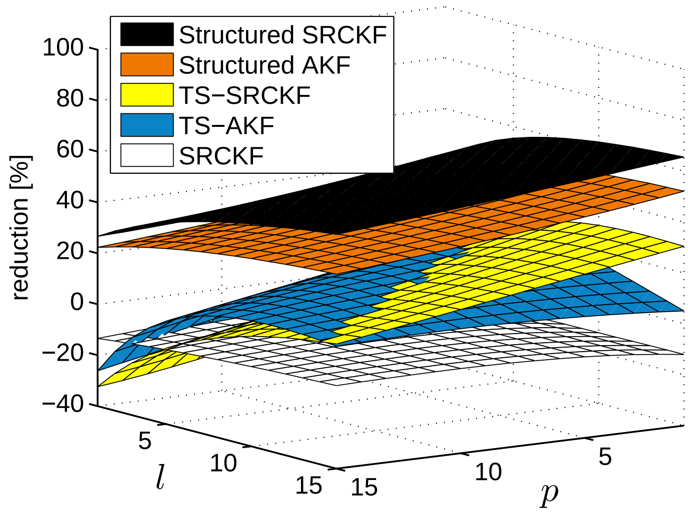


FIG. 1. *Comparison between the Structured SRCKF, Structured AKF, TS-SRCKF, TS-AKF and SRCKF as a function of p and l for a system with $n = 15$ states.*

TABLE 5

Parallel implementation of the Two-Stage SRCKF. The two dashed lines indicate that information needs to be exchanged (or data needs to be shared)

(48) S_k	$(\bar{R}_e, \bar{G}^x_k, \bar{S}^x_{k\|k})$ (53)
(49) $(R_e, G^\mu_k, \bar{S}^\mu_{k\|k})$	$(V_k, \bar{U}_{k+1})$ (54),(55)
(51) $\bar{S}^\mu_{k+1\|k}$	U_{k+1} (56)
(50) $\bar{\mu}_{k\|k}$	$\bar{x}_{k\|k}$ (57)
(52) $\bar{\mu}_{k+1\|k}$	$\bar{x}_{k+1\|k}$ (60)
	$\bar{S}^x_{k+1\|k}$ (58)

[12]. The performance of the new TS-SRCKF is exactly the same as that of the standard square-root covariance Kalman filter (SRCKF), i.e. both provide the MMSE error state estimate. However, the computational effort of the former, measured by the number of flops for one iteration, is usually much less than that of the SRCKF, even for problems of small size. Furthermore, it has been experimentally established that for some applications the new TS-SRCKF can require even less flops than the Hsiah's two-stage Kalman filter (TS-AKF).

Furthermore, similarly to what is claimed in [20], it has been experimentally demonstrated that the Structured AKF implementation, that basically makes use of the fact that the A-matrix of the augmented system has certain structure, requires even less flops than the two-stage implementations. And last, a new and faster implementation than all those mentioned above has also been derived in the paper. It is referred to as the Structured SRCKF as it is based on the SRCKF but makes use of the sparsity (structure) of the matrices.

REFERENCES

[1] A. T. Alouani, P. Xia, T. R. Rice, and W. D. Blair, *On the optimality of the two-stage state estimation in the presence of random bias*, IEEE Transactions on Automatic Control, 38:8(1993), pp. 1279–1282.

[2] G. J. Bierman, *The treatment of bias in the square-root information filter/smoother*, Journal of Optimization Theory and Applications, 16:1/2(1975)), pp. 165–178.

[3] Åke Björck, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996.

[4] M. Darouach and M. Zasadzinski, *Unbiased minimum variance estimation for systems with unknown exogenious inputs*, Automatica, 33:4(1997), pp. 717–719.

[5] B. Friedland, *Treatment of bias in recursive filtering*, IEEE Transactions on Automatic Control, AC-14:4(1969), pp. 359–367.

[6] B. Friedland, *On the properties of reduced-order Kalman filters*, IEEE Transactions on Automatic Control, 34:3(1989), pp. 321–324.

[7] F. Gustafsson, *Adaptive Filtering and Change Detection*, John Wiley & Sons Ltd., 2000.

[8] D. Haessig and B. Friedland, *Separate-bias estimation with reduced-order Kalman filters*, IEEE Transactions on Automatic Control, 43:7(1998), pp. 983–987.

[9] M. Hou and R. J. Patton, *Optimal filtering for systems with unknown inputs*, IEEE Transactions on Automatic Control, 43:3(1998), pp. 445–449.

[10] C.-S. Hsieh, *Robust two-stage Kalman filters for systems with unknown inputs*, IEEE Transactions on Automatic Control, 45:12(2000), pp. 2374–2378.

[11] C.-S. Hsieh, *General two-stage extended Kalman filters*, IEEE Transactions on Automatic Control, 48:2(2003), pp. 289–293.

[12] C.-S. Hsieh and F.-C. Chen, *Optimal solution of the two-stage Kalman estimator*, IEEE Transactions on Automatic Control, 44:1(1999), pp. 194–199.

[13] C.-S. Hsieh and F.-C. Chen, *General two-stage Kalman filters*, IEEE Transactions on Automatic Control, 45:4(2000), pp. 819–824.

[14] C.-S. Hsieh and F.-C. Chen, *Optimal minimal-order least-squares estimators via the general two-stage Kalman filter*, IEEE Transactions on Automatic Control, 46:11(2001), pp. 1772–1776.

[15] M. B. Ignagni, *An alternate derivation and extension of friedland's two-stage Kalman estimator*, IEEE Transactions on Automatic Control, 26:3(1981), pp. 746–750.

[16] M. B. Ignagni, *Separate-bias estimator with bias state noise*, IEEE Transactions on Automatic Control, 35:3(1990), pp. 338–341.

[17] M. B. Ignagni, *Optimal and suboptimal separate-bias Kalman extimators for a stochastic bias*, IEEE Transactions on Automatic Control, 45:3(2000), pp. 547–551.

[18] J.Ỹ. Keller and M. Darouach, *Optimal two-stage Kalman filter in the presence of random bias*, Automatica, 33:9(1997), pp. 1745–1748.

[19] P. K. Kitanidis, *Unbiased minimum-variance linear state estimation*, Automatica, 23(1987), pp. 775–778.

[20] K. Lo, Q. Lu, and W.H. Kwon, *Comments on "optimal solution of the two-stage Kalman*

estimator, IEEE Transactions on Automatic Control, 1(2002), pp. 198–199.

[21] J. M. Mendel, *Extension of friedland's bias filtering technique to a class of nonlinear systems*, IEEE Transactions on Automatic Control, 21(1976), pp. 296–298.

[22] J. M. Mendel and H. D. Washburn, *Multistage estimation of bias states in linear systems*, International Journal of Control, 28:4(1978), pp. 511–524.

[23] E. L. Shreve and W. R. Hendrick, *Separating bias and state estimates in a recursive second-order filter*, IEEE Transactions on Automatic Control, 19(1974), pp. 585–586.

[24] E. C. Tacker and C. C. Lee, *Liear filtering in the presence of time-varying bias*, IEEE Transactions on Automatic Control, 17(1972), pp. 828–829.

[25] A. Tanaka, *Parallel computation in linear discrete filtering*, IEEE Transactions on Automatic Control, 20(1975), pp. 573–575.

[26] M. Verhaegen and P. van Dooren, *Numerical aspects of different Kalman filter implementations*, IEEE Transactions on Automatic Control, 31:10(1986), pp. 907–917.

[27] M. Verhaegen and V. Verdult, *Filtering and System Identification: An Introduction*, TU-Delft, lecture notes for the course sc4040 (et4094) edition, 2003.

[28] A. Willsky and H. Jones, *A generalized likelihood ratioi approach to the detection and estimation of jumps in linear systems*, IEEE Transactions on Automatic Control, 1(1976), pp. 108–112.

[29] D. H. Zhou, Y. X. Sun, Y. G. Xi, and Z. J. Zhang, *Extension of friedland's separate-bias estimation to randomly time-varying bias of nonlinear systems*, IEEE Transactions on Automatic Control, 38:8(1993), pp. 1270–1273.