Computationally efficient optimal output decentralized estimation

André Bassong Onana, Mohamed Darouach, Michel Zasadzinski

COMPUTATIONALLY EFFICIENT OPTIMAL OUTPUT DECENTRALIZED ESTIMATION

A. BASSONG-ONANA , M. DAROUACH, M. ZASADZINSKI
CRAN - CNRS UA 821 - Université de Nancy I
186, rue de Lorraine - 54400 COSNES-ET-ROMAIN, France.

Abstract

This paper presents decentralized computational architectures for the optimal state estimation in stochastic large-scale linear systems. The main feature of the proposed architectures lies on elaborating each subsystem's decision not only by processing its own local data, but also by adjusting this decision with all other related subsystems local data. This adjustment procedure ensures the optimality of the decentralized filter. It is emphasized that the Kalman filter algorithm operates more efficiently when measurements are processed into low order subsets, especially when they are processed one at a time. Thus, using this feature in a decentralized scheme increases significantly computational savings and numerical stability. Architectures presented in this paper for the mechanization of decentralized estimators allow a high degree of parallelism and can be implemented on a wide range of computer networks.

1. Introduction

The main difficulties to overcome in processing data of large-scale interconnected structures deal with their high dimensionality. For this reason, reducing the dimension of large-scale systems has been ever since the great motivation of most investigators in designing schemes for estimation and control. Significant results obtained in small dimensioned systems are tried in large-scale systems through decompostion, with the hope of preserving the global performances such as optimality and stability.

Most previous works on decentralized estimation and control (Siljak and Vukcevic, 1978, Hodzic and Siljak, 1985, Stankovic and Siljak, 1989) consist of constructing the local estimators and controllers by using standard optimization schemes, thus treating inputs and outputs of each subsystem for its own estimator or controller. This strategy yields, in general, a loss of the global performances; in particular, the decentralized estimators developed by using this principle are globally suboptimal. The optimal decentralized estimators existing in the literature are built upon the exploration of all sensor observations for updating the estimation procedure. Two interesting approaches can be outlined.

For large-scale systems that are composed of geographically distributed sensors, with possible long distances, the attention was focused on constructing decentralized schemes that don't require the transmission of all sensor observations on a central location. By processing measurements locally on each sensing node (Hashemipour et al., 1988, Gardner and Leondes, 1990, Rao and Durant-Whyte, 1991), the need for large communication bandwidth is considerably reduced. But the global system model is treated as a whole on each node according equations

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + w(k) \\ z_i(k) = H_i x(k) + v_i(k) \end{cases}$$

Each subsystem computes the global Kalman filter equations related to its own observations. Since the processing time on each local processor is less than if all sensors were centralized, there is obviously a reduction of the global processing time when processors operate concurrently. However, despite the reduction of computation time, the global number of operations, as well as the global memory requirements, are enormously increased. Therefore, this approach falls in reducing the dimension of the global estimation problem. As it will be seen in section 2, each sensor can always be associated with a reduced-order system model, thus allowing a significant reduction of computational requirements.

The largely reported approach consists of transforming the global system model into a weakly or a strongly interconnected structure with, preferably, local outputs. The main algorithm using this approach was developed by Hassan et al. (1978). The optimal solution was obtained by performing successive orthogonalizations on the measurement subspaces, and Hassan et al. (1978) have shown that the numerical properties of the obtained decentralized filter are significantly better than the global

Kalman filter. However, as it will be shown later in this paper, the formulation of their algorithm could be considerably improved. Rhodes (1990) proposed a parallel scheme based on the decomposition of the steady-state Kalman filter into an observable canonical form. Despite the parallel structure for on-line implementations of the Kalman filter, the off-line computation of the global algebraic Riccati equation, necessary for the decomposition, incurs a risk of numerical instability.

The decentralized estimators developed in this paper are based on the output decentralization as described in section 2, and are applicable for both centralized or distributed sensors large-scale systems. In section 3, it is shown that the use of a recursive updating (RU) scheme for processing the estimation results in a considerable reduction of partial and global computational requirements. Two versions of the proposed decentralized filter, using the RU scheme, are presented in section 4, and three schemes are given in section 5 for their mechanization. One interesting feature of the resulting decentralized filter is that measurements can always be processed one at a time, regardless of the number of processors, thus improving the numerical behavior of the filter implementations. Computational aspects are also considered and, in section 6, the numerical stability of the decentralized filter is shown on an illustrative example.

2. Description of the system decomposition

In this section, we recall the output decomposition of large-scale systems described by a dynamical stochastic model ($\mathcal{S}$) of the following form

$$\begin{cases} \mathcal{X}(k+1) = \mathcal{A}\mathcal{X}(k) + \mathcal{B}u(k) + \mathcal{W}(k) \\ z(k) = \mathcal{H}\mathcal{X}(k) + v(k) \end{cases} \tag{1}$$

where $\mathcal{X}(k) \in \mathfrak{R}^{nx1}$, $u(k) \in \mathfrak{R}^{qx1}$, $z(k) \in \mathfrak{R}^{mx1}$ are the state, input and output vectors respectively, $\mathcal{X}(k)$ and $v(k)$ are the model noise and the measurement noise respectively, and matrices $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{H}$ have appropriate dimensions.

The output decentralization of the above system consists of transforming the system into a number of interconnected subsystems with local outputs. Naturally interconnected systems in which each subsystem has local sensors are not concerned by the following decomposition. Several techniques exist for the output decomposition of systems, but the one presented here is derived from the observable canonical form which provides a sparse stucture. So the main assumption we make is the observability of the pair ($\mathcal{H}$, $\mathcal{A}$), which implies its detectability (necessary condition for convergence and stability of the Kalman filter (De Souza et al., 1986)). Then, assuming that matrix $\mathcal{H}$ has full row rank, there exists a nonsingular matrix T which transforms system ($\mathcal{S}$) into system (S) given by

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + w(k) \\ z(k) = Hx(k) + v(k) \end{cases} \tag{2}$$

$$\text{with } A = T\mathcal{A}T^{-1} = \begin{bmatrix} A_{11} & A_{12} & \cdot\cdot & A_{1N} \\ A_{21} & A_{22} & \cdot\cdot & A_{2N} \\ \cdot & \cdot & \cdot\cdot & \cdot \\ A_{N1} & A_{N2} & \cdot\cdot & A_{NN} \end{bmatrix}, \quad B = T\mathcal{B} = [B_1^T \;\; B_2^T \;\; \ldots \; B_N^T]^T$$

$$A_{ii} = \begin{bmatrix} 0_{(n_i-m_i)1} & I_{n_i-m_i} \\ A_{ii}^1 & A_{ii}^2 \end{bmatrix} \in \mathfrak{R}^{n_i x n_i}, \quad A_{ij} = \begin{bmatrix} 0_{(n_i-m_j)n_j} \\ A_{ij}^2 \end{bmatrix} \in \mathfrak{R}^{n_i x n_j}, \; j \neq i, \; i=1,\ldots,m$$

$$H = \mathcal{B}T^{-1} = \mathrm{diag}(H_1,\ldots,H_N), \; H_i = [I_{m_i} \;\; 0_{m_i(n_i-m_i)}], \; x(k) = T\mathcal{X}(k) = [x_1^T(k) \;\; \ldots \; x_N^T(k)]^T,$$

$$w(k) = T\mathcal{W}(k) = [w_1^T(k) \;\; \ldots \; w_N^T(k)]^T, \; z(k) = [z_1^T(k) \;\; \ldots \; z_N^T(k)]^T, \; v(k) = [v_1^T(k) \;\; \ldots \; v_N^T(k)]^T$$

where $n_1 + \ldots + n_N = n$ and $m_1 + \ldots + m_N = m$. I_p and 0_{pr} are the identity matrix of order p and the null matrix of order pxr respectively.

Explicitely, the overall system (S) appears as N interconnected subsystems with local outputs, each subsystem (S_i) being described by the following equations

$$\begin{cases} x_i(k+1) = A_{ii}x_i(k) + \sum_{\substack{j=1 \\ j \neq i}}^{N} A_{ij}x_j(k) + B_i u(k) + w_i(k) \\ z_i(k) = H_i x_i(k) + v_i(k) \end{cases} \tag{3}$$

where all pairs (H_i, A_{ii}) are observable.

It must be noted that if $n_i = m_i$, then $A_{ii} = A_{ii}^1$ and $A_{ij} = A_{ij}^2$.

The construction of transformation T can be done in two steps

- first, construct the non singular matrix M which transforms system ($\mathcal{S}$) into its observable canonical form (Ogata, 1987), providing m subsystems;

- second, if the number of measurements is greater than the number N of available processors, then, by row and column permutations, group a given number of subsystems obtained in the previous step with respect to the canonical form, so as to match the number of available processors. If Q is the resulting permutation matrix, then T = QM.

Before developing the RU output decentralized filter, we describe in the following section the RU scheme for the global Kalman filtering.

3. The recursive updating Kalman filter

Consider the system model given by (2), and assume that noises $w(k)$ and $v(k)$ are white, gaussian, and satisfy

$$E\{w(k)\} = 0 \qquad E\{w(k)w^T(j)\} = W\delta(k,j),\ W \geq 0 \qquad E\{x(0)w^T(k)\} = 0 \qquad E\{w(k)v^T(k)\} = 0$$
$$E\{v(k)\} = 0 \qquad E\{v(k)v^T(j)\} = V\delta(k,j),\ V>0 \qquad E\{x(0)v^T(k)\} = 0.$$

Under the above assumption, it is well known that the following Kalman filter equations provide the optimal estimate of $x(k+1)$:

$$\hat{x}(0/0) = E\{x(0)\} \qquad P(0/0) = E\{(x(0)-\hat{x}(0/0))(x(0)-\hat{x}(0/0))^T\}$$
$$\hat{x}(k+1/k) = A\hat{x}(k/k) + Bu(k) \tag{4}$$
$$\hat{x}(k+1/k+1) = \hat{x}(k+1/k) + K_{(k+1)}^{op}(z(k+1)-H\hat{x}(k+1/k)) \tag{5}$$
$$K_{(k+1)}^{op} = P(k+1/k)H^T(HP(k+1/k)H^T+V)^{-1} \tag{6}$$
$$P(k+1/k) = AP(k/k)A^T + W \tag{7}$$
$$P(k+1/k+1) = (I-K_{(k+1)}^{op} H)P(k+1/k) \tag{8}$$

where $K_{(k+1)}^{op}$ is the Kalman gain. Computing this gain by classical matrix inversion algorithms may require too much computations and, even, give inaccurate numerical results. The objective in this section is to show how the measurements can be handled separately, in a recursive updating scheme, to produce the optimal estimate. For the purpose, the measurements are partitioned into r subsets (row partitioning of matrix H) such that

$$z_i(k) = H_i x(k) + v_i(k),\ i=1, ..., r \tag{9}$$

with

$$H = [H_1^T \ ... \ H_r^T]^T,\ z(k) = [z_1^T(k) \ ... \ z_r^T(k)]^T,\ v(k) = [v_1^T(k) \ ... \ v_r^T(k)]^T,\ H_i \in \Re^{m_i x n}$$

and

$$m_1 + m_2 + ... + m_r = m.$$

In the sequel, the measurement error covariance matrix V will be taken as block diagonal ($V=diag(V_1,...,V_r)$). This is not a limitation, since the orthogonal diagonalization of V can always be performed.

Let's define $\tilde{x}^i(k/k)$, $\tilde{K}^i(k)$, $\tilde{\varepsilon}^i(k/k)$, $\tilde{P}^i(k/k)$ as the partial estimate of $x(k)$, the partial gain, the partial estimation error and its covariance matrix respectively. The recursive updating Kalman filter equations are given by the following theorem.

Theorem 3.1

Let the system be described by model (2), and the measurements be partitioned according to equation (9). Subject to the statistical assumption given previously, then

a) the estimate of state at time k+1 can be obtained in r iterations according to the following recursions:

$$\tilde{x}^0(k+1/k+1) = A\hat{x}(k/k) + Bu(k) \tag{10}$$
$$\tilde{P}^0(k+1/k+1) = AP(k/k)A^T + W \tag{11}$$

$$\tilde{x}^{i}_{(k+1/k+1)} = \tilde{x}^{i-1}_{(k+1/k+1)} + \tilde{K}^{i}_{(k+1)}\left(z_{i(k+1)} - H_i\tilde{x}^{i-1}_{(k+1/k+1)}\right) \quad (12)$$

with

$$\tilde{K}^{i}_{(k+1)} = \tilde{P}^{i-1}_{(k+1/k+1)}\, H_i^T\left(H_i\tilde{P}^{i-1}_{(k+1/k+1)}H_i^T + V_i\right)^{-1} \quad (13)$$

$$\tilde{P}^{i}_{(k+1/k+1)} = (I - \tilde{K}^{i}_{(k+1)}\, H_i)\tilde{P}^{i-1}_{(k+1/k+1)} \quad (14)$$

At the last iteration, we have the optimality, i.e. $\hat{x}_{(k+1/k+1)} = \tilde{x}^{r}_{(k+1/k+1)}$ and $P_{(k+1/k+1)} = \tilde{P}^{r}_{(k+1/k+1)}$.

b) The optimal gain can be computed sequentially according to

$$\tilde{K}^{1|1}_{(k+1)} = \tilde{K}^{1}_{(k+1)} \quad (15)$$

$$\tilde{K}^{i|i}_{(k+1)} = \left[\left(I_n - \tilde{K}^{i}_{(k+1)}H_{i(k+1)}\right)\tilde{K}^{i-1|i-1}_{(k+1)} \;\middle|\; \tilde{K}^{i}_{(k+1)}\right],\ 1 < i \le r \quad (16)$$

which gives at the last iteration $K^{op}_{(k+1)} = \tilde{K}^{r|r}_{(k+1)}$, with $K^{op}_{(k+1)}$ defined by equation (6). ❐

The above theorem can be easily proved by induction and will, therefore, be omitted in this paper. As pointed out by Bierman (1973) (see also Mendel, 1971), the recursive updating scheme improves significantly the computational behaviour of the Kalman filter (see example in section 6), especially when measurements are processed one at a time. We exploit this feature to derive the improved output decentralized estimator.

4. Design of the RU output decentralized filter

Coming back to the decentralized Kalman filter developed by Hassan et al. (1978), there are some important points to be noted. The main drawback of their solution lies on the necessity of computing equations (8), (11) and (12) to obtain measurement updates. More precisely, these equations are dimensionally unconsistent when all subsystems don't have the same number of observations. Indeed, by using the same notations as in their paper (see Hassan et al. (1978) for more details), the expressions of the measurement residues $\tilde{y}^{s-1}_{s(k+1/k+1)}$ and $\tilde{y}^{s-2}_{s(k+1/k+1)}$, as defined in theorem 2, are given by :

$$\tilde{y}^{s-1}_{s(k+1/+1)} = y_{s(k+1)} - E\{y_{s(k+1)}|\mathbf{y}(k), y_{1(k+1)},\ldots,y_{s-1(k+1)}\}$$

$$\tilde{y}^{s-2}_{1(k+1/k+1)} = y_{s(k+1)} - E\{y_{s(k+1)}|\mathbf{y}(k), y_{1(k+1)},\ldots,y_{s-2(k+1)}\}$$

These residues can be expressed equivalently by :

$$\tilde{y}^{s-1}_{s(k+1/k+1)} = y_{s(k+1)} - H_s\tilde{x}^{l-1}_{l(k+1/k+1)} \text{ and } \tilde{y}^{s-2}_{s(k+1/k+1)} = y_{s(k+1)} - H_s\tilde{x}^{s-2}_{s(k+1/k+1)}.$$

It can be checked that the corresponding covariance matrices are :

$$P_{\tilde{y}^{s-1}_s\tilde{y}^{s-1}_s}(k+1/k+1) = H_s P_{\tilde{x}^{s-1}_s\tilde{x}^{s-1}_s}(k+1/k+1)H_s^T + V_s(k+1)$$

$$P_{\tilde{y}^{s-2}_s\tilde{y}^{s-2}_s}(k+1/k+1) = H_s P_{\tilde{x}^{s-2}_s\tilde{x}^{s-2}_s}(k+1/k+1)H_s^T + V_s(k+1)$$

$$P_{\tilde{y}^{s-2}_{s-1}\tilde{y}^{s-2}_{s-1}}(k+1/k+1) = H_{s-1} P_{\tilde{x}^{s-2}_{s-1}\tilde{x}^{s-2}_{s-1}}(k+1/k+1)H_{s-1}^T + V_{s-1}(k+1).$$

From the above equations, it is seen that matrix $K^{s-2}_{s_{s-1}}(k+1) = P_{\tilde{y}^{s-2}_s\tilde{y}^{s-2}_s}(k+1/k+1)P^{-1}_{\tilde{y}^{s-2}_{s-1}\tilde{y}^{s-2}_{s-1}}(k+1/k+1)$ in equation (12) can be computed only if matrices H_s and H_{s-1} have the same number of rows. It will be shown that the decentralized filter designed in this paper doesn't require the computation of equations (11) and (12) of Hassan et al. (1978). Therefore, the proposed filter is superior in the viewpoint of number of operations and numerical stability.

In this section, we assume that the output decentralization of the system has been achieved as outlined in section 2. However, the decentralized filter designed here applies also to the general case of any linear system which has an interconnected structure with independent local outputs (the pairs (H_i, A_{ii}) may not be in canonical form). Thus, each subsystem of such an interconnected structure is described by equations (3).

By analogy with the previous section, define

$\hat{x}_{i(k/k)}$, $\varepsilon_{i(k/k)}$: optimal estimate of $x_{i(k)}$ and its stimation error

$\tilde{x}^{r}_{i(k/k)}$, $\tilde{\varepsilon}^{r}_{i(k/k)}$: partial estimate of $x_{i(k)}$ with respect to local data of subsystem r, and its estimation error

$K_{i(k)}$, $\tilde{K}^{r}_{i(k)}$: the optimal and the partial gains of subsystem i

$P_{ij(k/k)} = E\{\varepsilon_{i(k/k)}\varepsilon^{T}_{j(k/k)}\}$ and $\tilde{P}^{r}_{ij(k/k)} = E\{\tilde{\varepsilon}^{r}_{i(k/k)}\tilde{\varepsilon}^{r\,T}_{j(k/k)}\}$ are the optimal and the partial interconnection error covariance matrices between subsystems i and j.

The recursive updating strategy for decentralizing the estimation problem consists of updating each local estimate by using, in a given order, successively the local data of each subsystem. When all local data have been explored, each resulting local estimate is optimal. The measurements may be explored globally in any order but, in the subsystems level, this order must be the same. Without loss of generality, we design decentralized schemes where measurements are explored from subsystem 1 to subsystem N.

Now, let us give the two versions of the decentralized filter. From the previous section, it is easy to show that the following equations provide the optimal decentralized state estimate:

Algorithm 4.1

Each subsystem i (i=1,...,N) proceeds as follows:

- Measurement updates initialization (time update):

$$\tilde{x}^{0}_{i(k+1/k+1)} = A_i\hat{x}_{(k/k)} + B_i u_{(k)} \tag{17}$$

$$\tilde{P}^{0}_{ii(k+1/k+1)} = A_i P_{(k/k)} A_i^{T} + W_{ii},\ i \le N \tag{18}$$

$$\tilde{P}^{0}_{i\bullet(k+1/k+1)} = A_i P_{(k/k)} A_{i\bullet}^{T} + W_{i\bullet},\ \tilde{P}^{0}_{\bullet i(k+1/k+1)} = \tilde{P}^{0\,T}_{i\bullet(k+1/k+1)},\ i \le N-1 \tag{19}$$

- Recursions on measurement updates (r=1, .., N):

$$\tilde{x}^{r}_{i(k+1/k+1)} = \tilde{x}^{r-1}_{i(k+1/k+1)} + \tilde{K}^{r}_{i(k+1)}(z_{r(k+1)} - H_r\tilde{x}^{r-1}_{r(k+1/k+1)}) \tag{20}$$

with

$$\tilde{K}^{r}_{i(k+1)} = \tilde{P}^{r-1}_{ir(k+1/k+1)} H_r^{T}(H_r\tilde{P}^{r-1}_{rr(k+1/k+1)}H_r^{T} + V_r)^{-1} \tag{21}$$

$$\tilde{P}^{r}_{ii(k+1/k+1)} = \tilde{P}^{r-1}_{ii(k+1/k+1)} - \tilde{K}^{r}_{i(k+1)} H_r\tilde{P}^{r-1\,T}_{ir(k+1/k+1)} \tag{22}$$

$$\tilde{P}^{r}_{i\bullet(k+1/k+1)} = \tilde{P}^{r-1}_{i\bullet(k+1/k+1)} - \tilde{K}^{r}_{i(k+1)} H_r\tilde{P}^{r-1}_{r/i\bullet(k+1/k+1)} \tag{23}$$

$$\tilde{P}^{r}_{\bullet i(k+1/k+1)} = \tilde{P}^{r\,T}_{i\bullet(k+1/k+1)} \tag{24}$$

At the last iteration, each local estimate is optimal, i.e. we have

$$\hat{x}_{i(k+1/k+1)} = \tilde{x}^{N}_{i(k+1/k+1)},\ P_{ii(k/k)} = \tilde{P}^{N}_{ii(k+1/k+1)},\ P_{i\bullet(k/k)} = \tilde{P}^{N}_{i\bullet(k+1/k+1)} \text{ and } P_{\bullet i(k/k)} = \tilde{P}^{N\,T}_{i\bullet(k+1/k+1)}$$

where

$A_i = [A_{i1} \dots A_{iN}]$, $A_{i\bullet} = [A_{i+1}^{T} \dots A_N^{T}]^{T}$, $W_{i\bullet} = [W_{i(i+1)} \dots W_{iN}]$, $P_{i\bullet(k/k)} = [P_{i(i+1)(k/k)} \dots P_{iN(k/k)}]$,

$P_{\bullet i(k/k)} = [P^{T}_{(i+1)i(k/k)} \dots P^{T}_{Ni(k/k)}]^{T}$, $\tilde{P}^{r}_{i\bullet(k/k)} = [\tilde{P}^{r}_{i(i+1)(k/k)} \dots \tilde{P}^{r}_{iN(k/k)}]$, $\tilde{P}^{r}_{\bullet i(k+1/k+1)} = [\tilde{P}^{r\,T}_{(i+1)i(k/k)} \dots \tilde{P}^{r\,T}_{Ni(k/k)}]^{T}$,

$\tilde{P}^{r}_{r/i\bullet(k/k)} = [\tilde{P}^{r}_{r(i+1)(k/k)} \dots \tilde{P}^{r}_{rN(k/k)}]$, with $i \le N-1$. ❐

Remark 4.1

1) When there are as processors as the number of measurements, only one measurement at a time is involved in the above equations. Equation (21) becomes

$$\tilde{K}^{r}_{i(k+1)} = \frac{\tilde{P}^{r-1}_{ir(k+1/k+1)} H_r^T}{H_r \tilde{P}^{r-1}_{rr(k+1/k+1)} H_r^T + V_r} \qquad (25)$$

Thus, a matrix inversion is reduced to a scalar inversion, and the roundoff error is considerably reduced over the global matrix inversion.

2) As it can be seen from equations (17)-(24), the decentralized filter design is simple. The interconnection covariances are assigned to each subsystem through matrix $\tilde{P}^{r}_{i.(k+1/k+1)}$. Their calculation can be done either by a central processor or, distributively, by each subsystem's local processor. It can be noted that equations (19) and (23) don't apply for i=N (no interconnection covariance is assigned to subsystem N).

3) Since the global error covariance matrix is symmetric, only upper block triangular terms, namely $\tilde{P}^{r}_{i.(k+1/k+1)}$, need to be computed. This has the double advantage of reducing calculations and increasing the numerical stability of the filter (the strict symmetry of the global error covariance matrix is preserved).

4) The partial terms $\Delta_{zr(k+1)} = z_{r(k+1)} - H_r \tilde{x}^{r-1}_{r(k+1/k+1)}$ and $M_{hr(k+1)} = H_r^T (H_r \tilde{P}^{r-1}_{rr(k+1/k+1)} H_r^T + V_r)^{-1}$ are common to all subsystems and are used for updating the estimates. The schemes presented in the next section, for the mechanization of the RU output decentralized filter, depend upon how these updating terms are handled for coordinating the estimation procedure.

5) Compared with the algorithm of Hassan et al. (1978), equations (17)-(19) and (21)-(23) above are the contraction of equations (4)-(6), (9), (10) and (13)-(15) of Hassan et al. (1978). One essential difference between both algorithms is that equation (20) of the above filter replaces equations (8), (11) and (12) of Hassan et al. (1978). In particular, it must be noted that two matrices have to be inverted, during each updating step, in their algorithm. ❐

Remark 4.2

The optimal gain $K^{op}_{(k+1)}$ can also be computed according to the above decentralized structure. Indeed, let $\tilde{K}^{r|r}_{(k+1)}$ denote the adjusted global gain at iteration r of the measurement updates. By writing $K^{op}_{(k+1)} = [K_1^{opT}{}_{(k+1)} \ \ldots \ K_N^{opT}{}_{(k+1)}]^T$ and $\tilde{K}^{r|r}_{(k+1)} = [\tilde{K}_1^{r|rT}{}_{(k+1)} \ \ldots \ \tilde{K}_N^{r|rT}{}_{(k+1)}]^T$, the following equations can be computed, within the recursions on measurement updates, for the global gain matrix:

$$\tilde{K}^{1|1}_{i(k+1)} = \tilde{K}^{1}_{i(k+1)} \qquad (26)$$

$$\tilde{K}^{r|r}_{i(k+1)} = \left[\tilde{K}^{r-1|r-1}_{i(k+1)} - \tilde{K}^{r}_{i(k+1)} H_r \tilde{K}^{r-1|r-1}_{r(k+1)} \mid \tilde{K}^{r}_{i(k+1)} \right], \ 1 < r \leq N \qquad (27)$$

and then $K^{op}_{i(k+1)} = \tilde{K}^{N|N}_{i(k+1)}$. ❐

It has been pointed out, in the above remarks, that only a scalar has to be inverted in equation (21) when there are as processors as the number of measurements. Since this is not the case in general, equation (21) must be treated as a matrix equation. In fact, algorithm 4.1 can always be modified so that one measurement at a time is processed by local processors during the updating cycle. Thus, the second version of the decentralized filter given below has a component-wise measurement updating procedure. The numerical improvement over the first version is obvious. The following equations of the modified decentralized filter are derived immediately from equations (17)-(24).

Algorithm 4.2

Each subsystem i (i=1,...,N) proceeds as follows:

- Measurement updates initialization (time update):

The same equations as in Algorithm 4.1.

- Recursions on measurement updates (r=1, .., N):

Equations (20)-(24) are equivalent to the following :

$$\tilde{x}_i^{r|0}(k+1/k+1) = \tilde{x}_i^{r-1}(k+1/k+1) \tag{28}$$

$$\tilde{P}_{ii}^{r|0}(k+1/k+1) = \tilde{P}_{ii}^{r-1}(k+1/k+1) \tag{29}$$

$$\tilde{P}_{i.}^{r|0}(k+1/k+1) = \tilde{P}_{i.}^{r-1}(k+1/k+1) \ , \quad \tilde{P}_{.i}^{r|0}(k+1/k+1) = \tilde{P}_{.i}^{r-1}(k+1/k+1) \tag{30}$$

Component-wise updating cycle ($s=1,...,m_r$):

$$\tilde{x}_i^{r|s}(k+1/k+1) = \tilde{x}_i^{r|s-1}(k+1/k+1) + \tilde{K}_i^{r|s}(k+1)(z_r^s(k+1) - H_r^s \tilde{x}_r^{r|s-1}(k+1/k+1)) \tag{31}$$

with

$$\tilde{K}_i^{r|s}(k+1) = \frac{\tilde{P}_{ir}^{r|s-1}(k+1/k+1)\, H_r^{s\,T}}{H_r^s \tilde{P}_{rr}^{r|s-1}(k+1/k+1) H_r^{s\,T} + V_r^s} \tag{32}$$

$$\tilde{P}_{ii}^{r|s}(k+1/k+1) = \tilde{P}_{ii}^{r|s-1}(k+1/k+1) - \tilde{K}_i^{r|s}(k+1)\, H_r^s \tilde{P}_{ir}^{r|s-1\,T}(k+1/k+1) \tag{33}$$

$$\tilde{P}_{i.}^{r|s}(k+1/k+1) = \tilde{P}_{i.}^{r|s-1}(k+1/k+1) - \tilde{K}_i^{r|s}(k+1)\, H_r^s \tilde{P}_{r/i.}^{r|s-1}(k+1/k+1) \tag{34}$$

$$\tilde{P}_{.i}^{r|s}(k+1/k+1) = \tilde{P}_{i.}^{r|s\,T}(k+1/k+1) \tag{35}$$

and $$\tilde{x}_i^{r}(k+1/k+1) = \tilde{x}_i^{r|m_r}(k+1/k+1) \tag{36}$$

$$\tilde{P}_{ii}^{r}(k+1/k+1) = \tilde{P}_{ii}^{r|m_r}(k+1/k+1) \tag{37}$$

$$\tilde{P}_{i.}^{r}(k+1/k+1) = \tilde{P}_{i.}^{r|m_r}(k+1/k+1) \ , \ \tilde{P}_{.i}^{r}(k+1/k+1) = \tilde{P}_{i.}^{r\,T}(k+1/k+1) \tag{38}$$

where $z_r(k+1) = [z_r^1(k+1) \ ... \ z_r^{m_r}(k+1)]^T$, $H_r = [H_r^1 \ ... \ H_r^{m_r}]^T$ and $V_r = \mathrm{diag}(V_r^1, ..., V_r^{m_r})$. $\tilde{x}_i^{r|s}(k+1/k+1)$, $\tilde{K}_i^{r|s}(k+1)$, $\tilde{P}_{ii}^{r|s}(k+1/k+1)$ and $\tilde{P}_{i.}^{r|s}(k+1/k+1)$ are the partial estimate of state, the partial gain, the partial local and interconnection covariance matrices respectively, with respect to measurement $z_r^s(k+1)$. In this case, the updating terms are $\Delta_{zr}^{r|s}(k+1) = z_r^s(k+1) - H_r^s \tilde{x}_r^{r|s-1}(k+1/k+1)$ and $M_{zr}^{r|s}(k+1) = H_r^s \tilde{P}_{rr}^{r|s-1}(k+1/k+1) H_r^{s\,T} + V_r^s$. ❐

Remark 4.3

Since the objective in the filter implemtations is to reduce as far as possible the computational requirements, in writing computer programs for algorithms 4.1 and 4.2, one may exploit the sparse structure of the decomposed system to suppress extra useless calculations. Indeed, most computations are involved in the time update equations. These equations can be written in a reduced order form by noting that matrix A_i ($i \le N$) has the form

$$A_i = \left[\begin{array}{c|c|c} 0 & I_{n_i - m_i} & 0 \\ \hline & A_i^2 & \end{array}\right] \tag{39}$$

5. Mechanization of the RU output decentralized filter

In this section, we discuss some aspects of the output decentralized filter implementation. As pointed out in the previous section, the strategy for the implementation of the decentralized filter depends on how the updating terms are treated. In general, the architecture adopted will depend on the hardware (and even software) available for each application. One advantage of the architectures presented below is their adaptability with a wide range of existing computer networks. In the sequel,

we give two multi-level computational architecures and describe the information flow within these networks.

Coordination by a super processor

The most natural architecture for processing the decentralized filter is undoubtably the hierarchical architecture supported by a central processor (or super processor). All outputs measurements are passed to the super processor for computing the updating terms Δ_{zr} and M_{hr} (or Δ_{zr}^{rls} and M_{zr}^{rls}) and the interconnection covariance matrix $\tilde{P}_{i.}^{r}$ (or $\tilde{P}_{i.}^{rls}$), which are then passed back to the local processors to compute $\tilde{K}_i^r$, $\tilde{P}_{i,i}^r$ and $\tilde{x}_i^r$ (or $\tilde{K}_i^{rls}$, $\tilde{P}_{i,i}^{rls}$ and $\tilde{x}_i^{rls}$). Since no interconnection covariance term is assigned to the last local processor N, this processor can be supressed, and its computations supported by the super processor, thus removing the need of an additional processor. Therefore, in the architecture of figures 1a and 1b, showing how the super processor and the local processors are interconnected, the last local processor may be processor N or N-1.

The architecture in Fig. 1a is more suitable for centralized sensors large-scale systems. Indeed, for distributed sensors large-scale systems, it is preferable to process measurements locally in order to avoid transmission of all observations to the super processor. In the architecture of figure 1.b below, each local processor is charged of its related updating terms which it sends directly, through a bidirectional transmission bus, to all others local processors to adjust their estimates. Thus, the super processor is discharged from the calculation of Δ_{zr} and M_{hr} (or Δ_{zr}^{rls} and M_{zr}^{rls}); it just has to compute and transmit interconnection covariance matrices. It may also send an indicator to inform a given processor about the transmission of updating terms.

The main drawback of hierarchical architectures is, obviously, their dependency on the central processor. The efficiency of such structures requires a high reliability of the super processor. Therefore, the processing capability and the power of the super processor, and even the fluidity of exchanges between subsystems and the super processor, are required.

Coordination by shared memory

Communication between subsystems during the updating cycles can be carried out via shared memory. Indeed, in the architecture of figure 2, a shared memory serves as a bidirectional 'mailbox' between subsystems through a suitable logical interface and a hardware bus interface. The existence of memories that allow simultaneous reading and storage by one processor at a time legitimates this third architecture.

In this configuration, each subsystem achieves the complete updating procedure by reading, from the shared memory, the updating terms issued by one of the other subsystems. In turn, each subsystem stores, in the shared memory, the updating terms for the next updating cycle. At the end of the procedure, that is when the optimal estimates have been attained in the subsystems level, each local processor sends its local results to the reconstitution unit. The latter reconstitutes the global estimate $\hat{x}_{(k+1/k+1)}$ and the global error covariance matrix $P_{(k+1/k+1)}$, and gives them back to subsystems for the next estimation step. It can be noted that all computations are supported by subsystems. The shared memory and the reconstitution unit form two levels of coordination, the measurement-coordination and the time-coordination respectively. In fact, the reconstitution unit may be another shared memory. Thus, compared to the previous hierarchical architectures, architecture of figure 2 increases the degree of parallelism. A high degree of parallelism can be obtained by eliminating memory access conflicts between local processors.

For each version of the RU decentralized filter, the global numbers of elementary operations are the same in the above architectures; only the subsystems numbers of operations are sensibly different. In the following paragraph, we show that the reduction of computational requirements is substantial in the subsystems level, and even notable for the overall system.

Study of computational requirements

Since the computational requirements (processing time and memory requirements) depend on each processor and/or the multiprocessor environment, we do not intend to draw exact numerical conclusions in this study. However, the number of elementary operations (additions and multiplications) gives a good measure of these requirements, and will here be used to give an idea of computational savings in the decentralized filter implementations. By taking into account all eventual symmetries of matrices, the global numbers of elementary operations have the following expressions.

For multiplications we have:

- conventional Kalman filter :

$$\frac{1}{2} nm(5n+3m+6) + \frac{1}{2} n^2(3n+1) + \frac{1}{2}(m-1)(m^2+2m+2) + (n(n+q))$$

- sequential Kalman filter :

$$\frac{1}{2} n(5n+9) + \frac{1}{2} n^2(3n+1) + (n+q) + n(m-1)(n+4)$$

- Algorithm 4.1

$$\sum_{i=1}^{N} \Big(\frac{1}{2} m_i n_i (2n_i+3m_i+3) + \frac{1}{2}(m_i-1)(m_i^2+m_i+2) + \frac{1}{2} n n_i (2n+n_i+1) + n_i(n+q) + \sum_{r=1}^{N} (m_r(n_r+n_i) + n_i n_r (n_i+3m_r+3)) \Big) + \sum_{i=1}^{N-1} \Big(n_i n^2 + n n_i s_i + \sum_{r=1}^{N} n_i n_r (m_r+s_i-1) \Big)$$

- Algorithm 4.2

$$\sum_{i=1}^{N} \Big(m_i n_i (n_i+3) + n_i(n+m+q) + \frac{1}{2} n n_i (2n+n_i+1) + \frac{1}{2} n_i (n_i+5) t_N \Big) + \sum_{i=1}^{N-1} \Big(n_i n (n+s_i) + n_i (s_i+1) t_N \Big)$$

and for additions:

- conventional Kalman filter

$$\frac{1}{2} nm(5n+3m) + \frac{1}{2} m(m-1)^2 + \frac{1}{2} n^2(3n-1) + n(n+q-1)$$

- sequential Kalman filter

$$\frac{1}{2} n(5n+3) + \frac{1}{2} n^2(3n-1) + n(n+q-1) + \frac{1}{2}(m-1)(3n^2+7n-7)$$

- Algorithm 4.1

$$\sum_{i=1}^{N} \Big(\frac{1}{2} m_i n_i (2n_i+3m_i-1) + \frac{1}{2} m_i (m_i-1)^2 + \frac{1}{2} n n_i (2n+n_i-1) + n_i(n+q-1) + \sum_{r=1}^{N} (m_r n_r (n_i+1) + n_i n_r (n_i+2m_r-1)) \Big) + \sum_{i=1}^{N-1} \Big(n_i n(n-1) + n n_i s_i + \sum_{r=1}^{N} n_i n_r (m_r+s_i-1) \Big)$$

- Algorithm 4.2

$$\sum_{i=1}^{N} \Big(m_i n_i (n_i+1) + n_i(n+q-1) + \frac{1}{2} n n_i (2n+n_i-1) + \frac{1}{2} n_i (n_i+3) t_N \Big) + \sum_{i=1}^{N-1} \Big(n_i n (n+s_i-1) + n_i s_i t_N \Big)$$

where $s_i = \sum_{j=i+1}^{N} n_j$ and $t_N = \sum_{r=1}^{N} n_r m_r$.

The above established numbers of operations have been used to compare the global and the decentralized filters in Fig. 3a, and algorithms 4.1 and 4.2 in Fig. 3b, when the subsystems have the same dimensions ($n_i = \frac{n}{N}$, $m_i = \frac{m}{N}$). In the case of a component-wise decomposition of outputs (i.e. the number of subsystems equals the number of outputs), the study of the global numbers of operations in terms of the number of measurements shows that, for a fixed number of state variables, the computational savings increase with the number of measurements. In fact, these savings become

significative when $m>\frac{n}{2}$. It can be mentioned that considerable savings can be obtained with the sequential Kalman filter but, as it is shown in the example, this advantage falls when the numerical stability is of concern. We have not plotted the numbers of operations in the subsystems level, since the gain of operations over the global implementation is obviously substantial. It can be noted that the number of elementary operations executed by local processors decreases when the number of subsystems increases. On Fig. 3, DKF1, DKF2, RUKF and CKF stand for Algorithm 4.1, Algorithm 4.2, RU Kalman filter and conventional Kalman filter respectively.

6. Numerical example

The superiority of the proposed decentralized filters on the other algorithms has been confirmed by many simulation examples. To compare the numerical behaviour of the algorithms considered in this paper, they have been applied, on MATLAB 386, to a linearized model of the system formed by an interconnection of eleven synchronous machines and given in Hassan et al. (1978). This model has the form of equations (1) which matrices are given in Appendix. The equivalent interconnected system satisfies equations (2)-(3) which, in this example, correspond to an interconnection of 10 subsystems having two states and one observation. For lack of place in this paper, we do not give explicitely the transformed system matrices (they can easily be derived by transformation matrix T given in Appendix). It must be noted here that the conditions for the convergence of the Kalman filter to a stable steady-state filter (De Souza et al., 1986) are satisfied (the pair ($\mathcal{H}$, $\mathcal{A}$) is observable, and the pair ($\mathcal{A}$, $\mathcal{W}^{\frac{1}{2}}$) is reachable).

Implementation on a 10 local processors system

Figures 4 and 5 below, plotting the evolution of the trace of the global error covariance matrix, show that the standard Kalman filter diverges, while the others converge. Indeed, from Fig.4 , it is seen that after the 40^{th} iteration, the computed error covariance matrix becomes very large for the standard Kalman filter. Although system matrices are well-conditioned (cond($\mathcal{A}$)=3.2620, cond(A)=8.9121, and cond($\mathcal{H}$)=cond(H)=1), the effects of round-off error propagation are more desastrous for the conventional Kalman filter, proving its numerical instabilty. On the other hand Fig. 5 shows that the sequential Kalman filter, as well as all versions of the decentralized filter, are more numerically stable than the conventional Kalman filter, thus improving convergence and stability conditions when the system has large dimension. This conclusion is exact each subsystem has a scalar measurement, however it is not always the case when, unfortunately, less processors than the number of measurements are available.

Implementation on a 6 local processors system

If only 6 local processors are available, it is always possible, by means of row and column permutations, to fuse certain subsystems with respect to the observable canonical form, as pointed out in section 2. In this example, let's fuse subsystems (H_1, A_{11}) and (H_2, A_{22}), subsystems (H_5, A_{55}) and (H_6, A_{66}), and subsystems (H_8, A_{88}), (H_9, A_{99}) and (H_{10}, A_{1010}). This is achieved by the permutation matrix Q given in Appendix. The new transformed system is an interconnection of 6 subsystems with greater dimensions given by :

$$\begin{cases} n_1=4 \\ m_1=2 \end{cases} \quad \begin{cases} n_2=2 \\ m_1=1 \end{cases} \quad \begin{cases} n_3=2 \\ m_3=1 \end{cases} \quad \begin{cases} n_4=4 \\ m_4=2 \end{cases} \quad \begin{cases} n_5=2 \\ m_5=1 \end{cases} \quad \begin{cases} n_6=6 \\ m_6=3 \end{cases}$$

The implemention of the above algorithms reveals the superiority of Algorithm 4.2 of the decentralized filter. Indeed, it can be seen in this case, that the sequential Kalman filter (Fig. 6) and the decentralized filter in Algorithm 4.1 (Fig. 7) diverge numerically, while Algorithm 4.2 still gives good results (Fig. 8). Therefore, when the computation of equation (20) requires a matrix inversion, Algorithm 4.1 is more sensitive to round-off errors (which effects are amplified when matrix M_{hr} has large dimension) than Algorithm 4.2. In particular, it can be noted that the sequential Kalman filter is numerically interesting only for a suitable partitionning of measurements (Mendel, 1971).

Conclusion

In this paper, computationally efficient output decentralized filters have been developed. Particular attention has been focused on numerical aspects, as well as the implementation of the

proposed algorithms on computer systems. Various schemes presented give enough informations for the application of the output decentralized estimation. Architectures presented increase the performances and reduce computational requirements for the global Kalman filter implementation. By means of the output decentralization, the problem of optimal decentralized estimation is solved both for distributed and centralized sensors systems. The maximal performances are obtained when measurements are processed one at a time. Since this feature is preserved whatever is the number of processors, through Algorithm 4.2, and since this number is not a limitation in the implementation of the decentralized schemes presented, these schemes are widely applicable without loss of performances.

References

BIERMAN, G.J., 1973, A comparison between linear filtering algorithms. *I.E.E.E. Transactions on Aerospace and Electronic Systems*, **9**, 28-37.

DE SOUZA, C.E., GEVERS, M.R., and GOODWIN, G.C., 1986. Riccati equations in optimal filtering of nonstabilizable systems having singular state transition matrices. *I.E.E.E. Transactions on Automatic Control*, **31**, 831-837.

FOULARD, C., GENTIL, S., and SANDRAZ, J.P., 1987. Commande et régulation par calculateur numérique. *Eyrolles*, Paris, France.

GARDNER, W.T., and LEONDES, C.T., 1990, Gain transfert : an algorithm for decentralized hierarchical estimation. *International Journal of Control*, **52**, 279-292.

HASHEMIPOUR, H.R., ROY, S., and LAUB, A.J., 1988, Decentralized structures for parallel Kalman filtering. *I.E.E.E. Transactions on Automatic Control*, **33**, 88-94.

HASSAN, M.F., SALUT, G., SINGH, M.G., and TITLI, A., 1978, A decentralized computational algorithm for the global Kalman filter. *I.E.E.E. Transactions on Automatic Control*, **23**, 262-268.

HODZIC, M., and SILJAK, D.D., 1985, Estimation and control of large sparse systems. *Automatica*, **21**, 277-292.

MENDEL, J.M., 1971, Computational requirements for a discrete Kalman filter. *I.E.E.E. Transactions on Automatic Control*, **16**, 748-758.

OGATA, K., 1987. Discrete-time control systems. *Prentice-Hall, Englewood Cliffs*, New Jersey.

RAO, B.S., and DURRANT-WHYTE, H.F., 1991, Fully decentralised algorithm for multisensor Kalman filtering. *I.E.E. Proceedings-D*, **138**, 413-420.

RHODES, I.B., 1990, A parallel decomposition for Kalman filters. *I.E.E.E. Transactions on Automatic Control*, **36**, 322-327.

SILJAK, D.D., and VUKCEVIC, M.B., 1978, On decentralized estimation. *International Journal of Control*, **27**, 113-131.

STANKOVIC, S., and SILJAK, D.D., 1989, Sequential LQG optimization of hierarchically structured systems. *Automatica*, **25**, 545-559.

Appendix

For the system described in the previous example, which satisfies equations (1), the matrices are given by:

$$\mathcal{H} = \begin{bmatrix} \mathcal{H}_1 \\ \mathcal{H}_2 \end{bmatrix} \text{ and } \mathcal{A} = \begin{bmatrix} \mathcal{A}_{11} & \mathcal{A}_{12} \\ \mathcal{A}_{21} & \mathcal{A}_{22} \\ \mathcal{A}_{31} & \mathcal{A}_{32} \end{bmatrix} \text{ with}$$

$$\mathcal{H}_1 = \begin{bmatrix} 0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0 \end{bmatrix}$$

$$\mathcal{H}_2 = \begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1 \end{bmatrix}$$

$$\mathcal{A}_{11} = \begin{bmatrix} 1 & 0.0063 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.5870 & 1 & 0.0546 & 0 & 0.0336 & 0 & 0.0164 & 0 & 0.0254 & 0 \\ 0 & 0 & 1 & 0.0053 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0990 & 0 & -0.8772 & 1 & 0.0600 & 0 & 0.0875 & 0 & 0.0417 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0.0063 & 0 & 0 & 0 & 0 \\ 0.0558 & 0 & 0.0641 & 0 & -0.7250 & 1 & 0.0389 & 0 & 0.0269 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0.0063 & 0 & 0 \\ 0.0532 & 0 & 0.01179 & 0 & 0.0478 & 0 & -0.8246 & 1 & 0.0254 & 0 \end{bmatrix}$$

$$\mathcal{A}_{21} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0.0063 \\ 0.0706 & 0 & 0.0537 & 0 & 0.0374 & 0 & 0.0269 & 0 & -0.7790 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0123 & 0 & 0.0122 & 0 & 0.0105 & 0 & 0.0075 & 0 & 0.0107 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0440 & 0 & 0.0538 & 0 & 0.0356 & 0 & 0.0474 & 0 & 0.0344 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0012 & 0 & -0.0019 & 0 & 0.0004 & 0 & -0.0024 & 0 & 0.0026 & 0 \end{bmatrix}$$

$$\mathcal{A}_{31} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0317 & 0 & 0.0467 & 0 & 0.0248 & 0.0248 & 0 & 0.0350 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0099 & 0 & 0.0087 & 0 & 0.0060 & 0.0060 & 0 & 0.0050 & 0 & 0 \end{bmatrix}$$

$$\mathcal{A}_{12} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.0103 & 0 & -0.0169 & 0 & -0.0300 & 0 & -0.0698 & 0 & -0.0786 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.5900 & 0 \\ 0.0167 & 0 & 0.0095 & 0 & -0.0125 & 0 & -0.0440 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.0720 & 0 \\ 0.0026 & 0 & -0.0066 & 0 & -0.0293 & 0 & -0.0734 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.0640 & 0 \\ 0.0166 & 0 & 0.0072 & 0 & -0.0190 & 0 & -0.0478 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathcal{A}_{22} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.0560 & 0 \\ 0.0162 & 0 & -0.0004 & 0 & -0.0081 & 0 & -0.0590 & 0 & 0 & 0 \\ 1 & 0.0063 & 0 & 0 & 0 & 0 & 0 & 0 & -0.0620 & 0 \\ -0.6230 & 1 & 0.0121 & 0 & -0.0058 & 0 & -0.0570 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0.0063 & 0 & 0 & 0 & 0 & -0.2750 & 0 \\ 0.0736 & 0 & -1.0420 & 1 & 0.0350 & 0 & 0.0450 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0.0063 & 0 & 0 & -0.0536 & 0 \\ 0.0078 & 0 & -0.0043 & 0 & -0.5580 & 1 & -0.0420 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathcal{A}_{32} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0.0063 & 0 & 0 \\ 0.0380 & 0 & 0.0455 & 0 & 0.0500 & 0 & -0.8820 & 1 & -0.0029 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0.0063 \\ 0.0098 & 0 & 0.0033 & 0 & 0.0100 & 0 & -0.0063 & 0 & -0.6880 & 1 \end{bmatrix}$$

and the noises covariance matrices are V = I and $\mathcal{W}$= 5I, with initial conditions:

$\mathcal{P}(0/0) = 25\ \mathrm{I}$, $\hat{x}(0/0) = 10\ [1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1]^{T}$

The canonical form transformation T outlined in section 2 is given by

$$\mathrm{T} = \begin{bmatrix} \mathrm{T}_{11} & \mathrm{T}_{12} \\ \mathrm{T}_{21} & \mathrm{T}_{22} \\ \mathrm{T}_{31} & \mathrm{T}_{32} \end{bmatrix} \quad \text{with}$$

$$T_{11} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.5870 & 1 & 0.0546 & 0 & 0.0336 & 0 & 0.0164 & 0 & 0.0254 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0990 & 0 & -0.8772 & 1 & 0.0600 & 0 & 0.0875 & 0 & 0.0417 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0.0658 & 0 & 0.0641 & 0 & -0.7250 & 1 & 0.0389 & 0 & 0.0269 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0.0532 & 0 & 0.01179 & 0 & 0.0478 & 0 & -0.8246 & 1 & 0.0254 & 0 \end{bmatrix}$$

$$T_{21} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0.0063 \\ 0.0706 & 0 & 0.0537 & 0 & 0.0374 & 0 & 0.0269 & 0 & -0.7790 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0123 & 0 & 0.0122 & 0 & 0.0105 & 0 & 0.0075 & 0 & 0.0107 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0440 & 0 & 0.0538 & 0 & 0.0356 & 0 & 0.0474 & 0 & 0.0344 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0012 & 0 & -0.0019 & 0 & 0.0004 & 0 & -0.0024 & 0 & 0.0026 & 0 \end{bmatrix}$$

$$T_{31} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0317 & 0 & 0.0467 & 0 & 0.0248 & 0 & 0.0350 & 0 & 0.0251 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0099 & 0 & 0.0087 & 0 & 0.0060 & 0 & 0.0050 & 0 & 0.0113 & 0 \end{bmatrix}$$

$$T_{12} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.0103 & 0 & -0.0169 & 0 & -0.0300 & 0 & -0.0698 & 0 & -0.0786 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0167 & 0 & 0.0095 & 0 & -0.0125 & 0 & -0.0440 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0026 & 0 & -0.0066 & 0 & -0.0293 & 0 & -0.0734 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0166 & 0 & 0.0072 & 0 & -0.0190 & 0 & -0.0478 & 0 & 0 & 0 \end{bmatrix}$$

$$T_{22} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0162 & 0 & -0.0004 & 0 & -0.0081 & 0 & -0.0590 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.6230 & 1 & 0.0121 & 0 & -0.0058 & 0 & -0.0570 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0736 & 0 & -1.0420 & 1 & 0.0350 & 0 & 0.0450 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0.0078 & 0 & -0.0043 & 0 & -0.5580 & 1 & -0.0420 & & 0 & 0 \end{bmatrix}$$

$$T_{32} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0.0380 & 0 & 0.0455 & 0 & 0.0500 & 0 & -0.8820 & 1 & -0.0029 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0.0098 & 0 & 0.0033 & 0 & 0.0100 & 0 & -0.0063 & 0 & -0.6880 & 1 \end{bmatrix}$$

and the permutation matrix used to form the specified subsystems fusion is

$$Q = \begin{bmatrix} Q_1^T & Q_2^T & Q_3^T & Q_4^T \end{bmatrix}^T \text{ with}$$

$$Q_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$Q_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$Q_3 = \begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0 \end{bmatrix}$$

$$Q_4 = \begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1 \end{bmatrix}$$

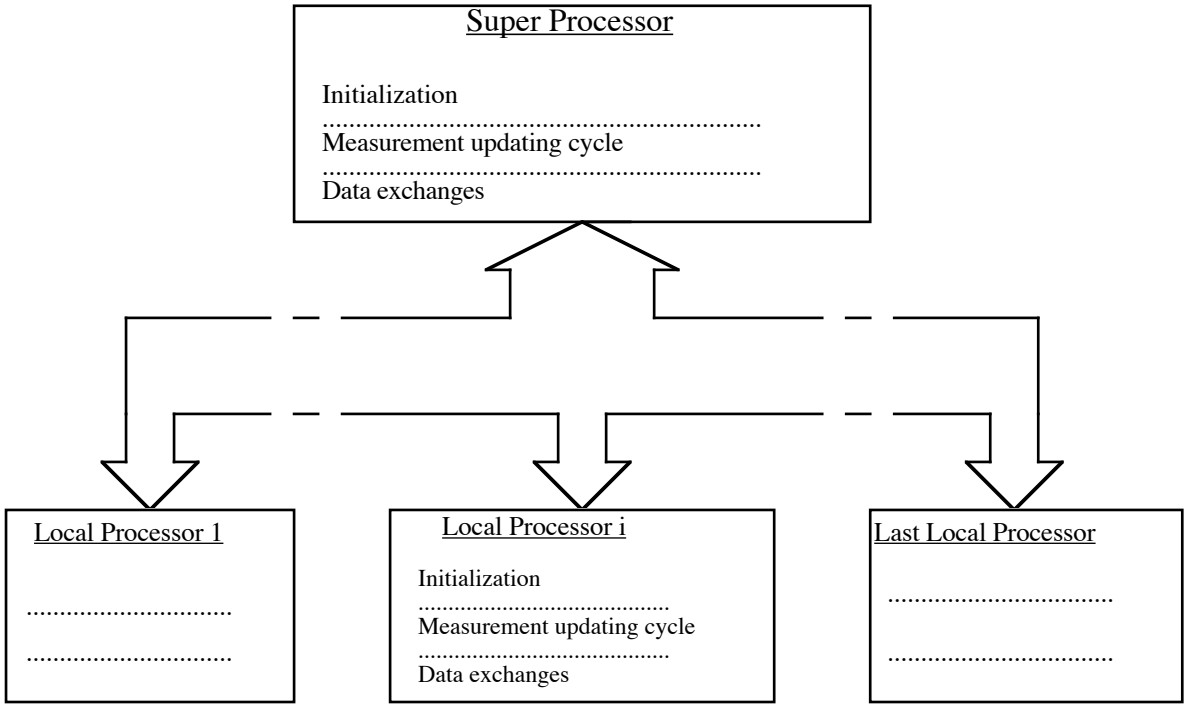

Fig.1a: Hierarchical architecture with dedicated super processor

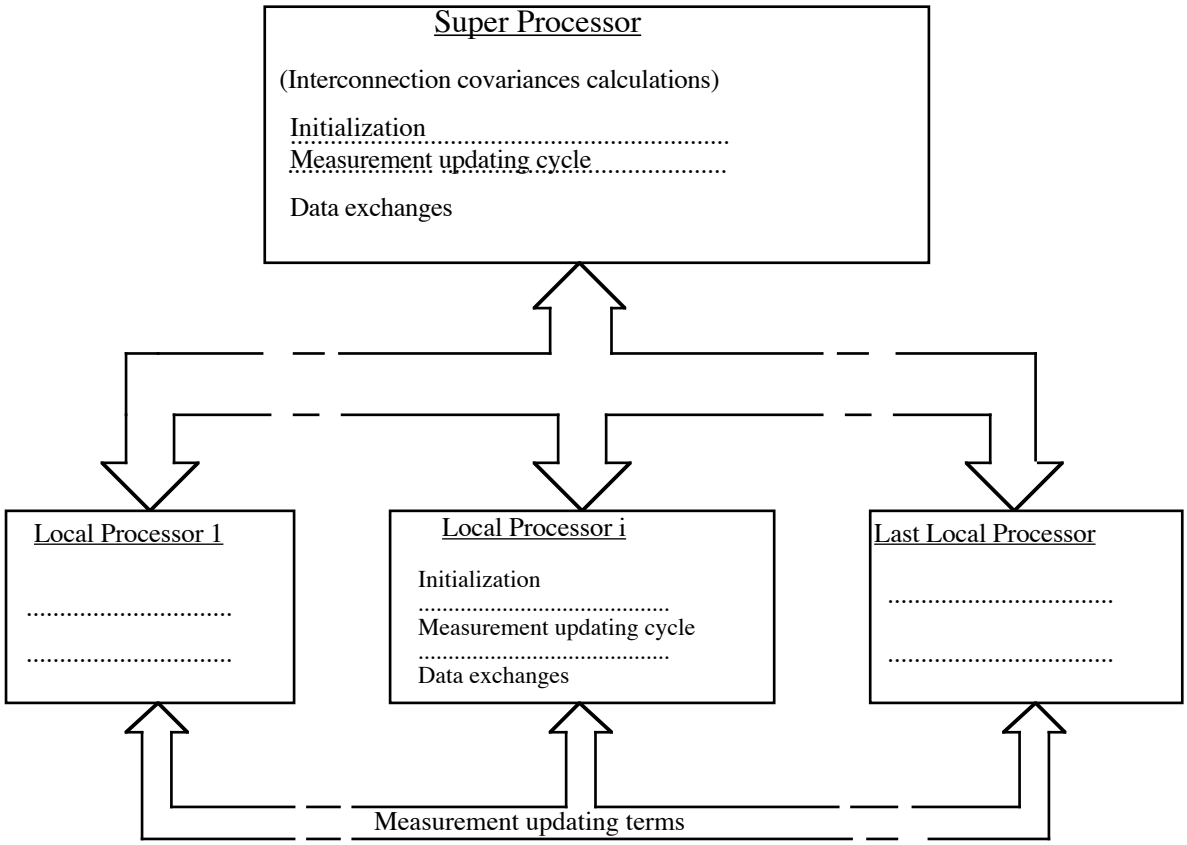

Fig.1b: Direct communication between local processors

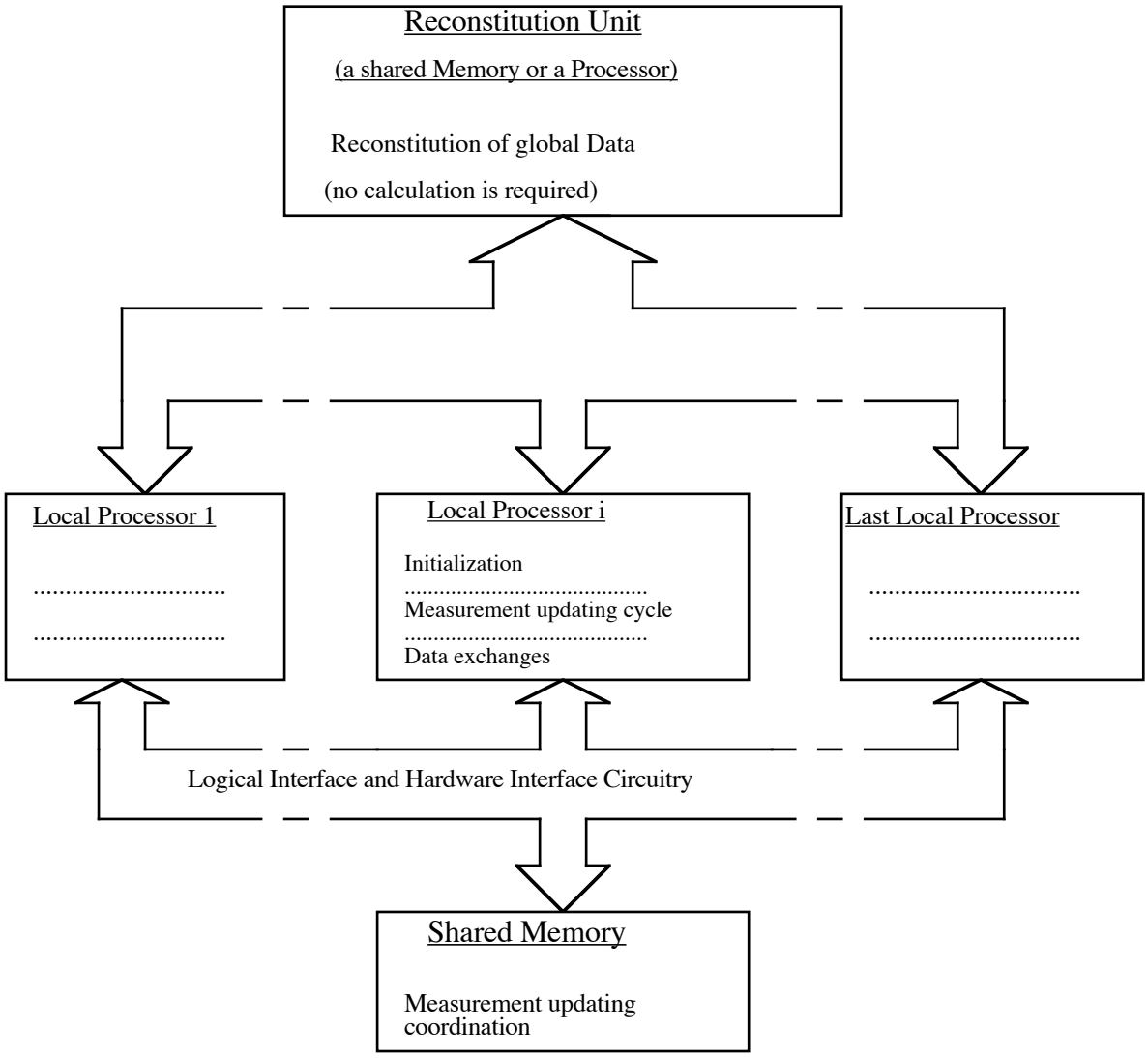

Fig.2: A shared memory coordination for the decentralized filtering.

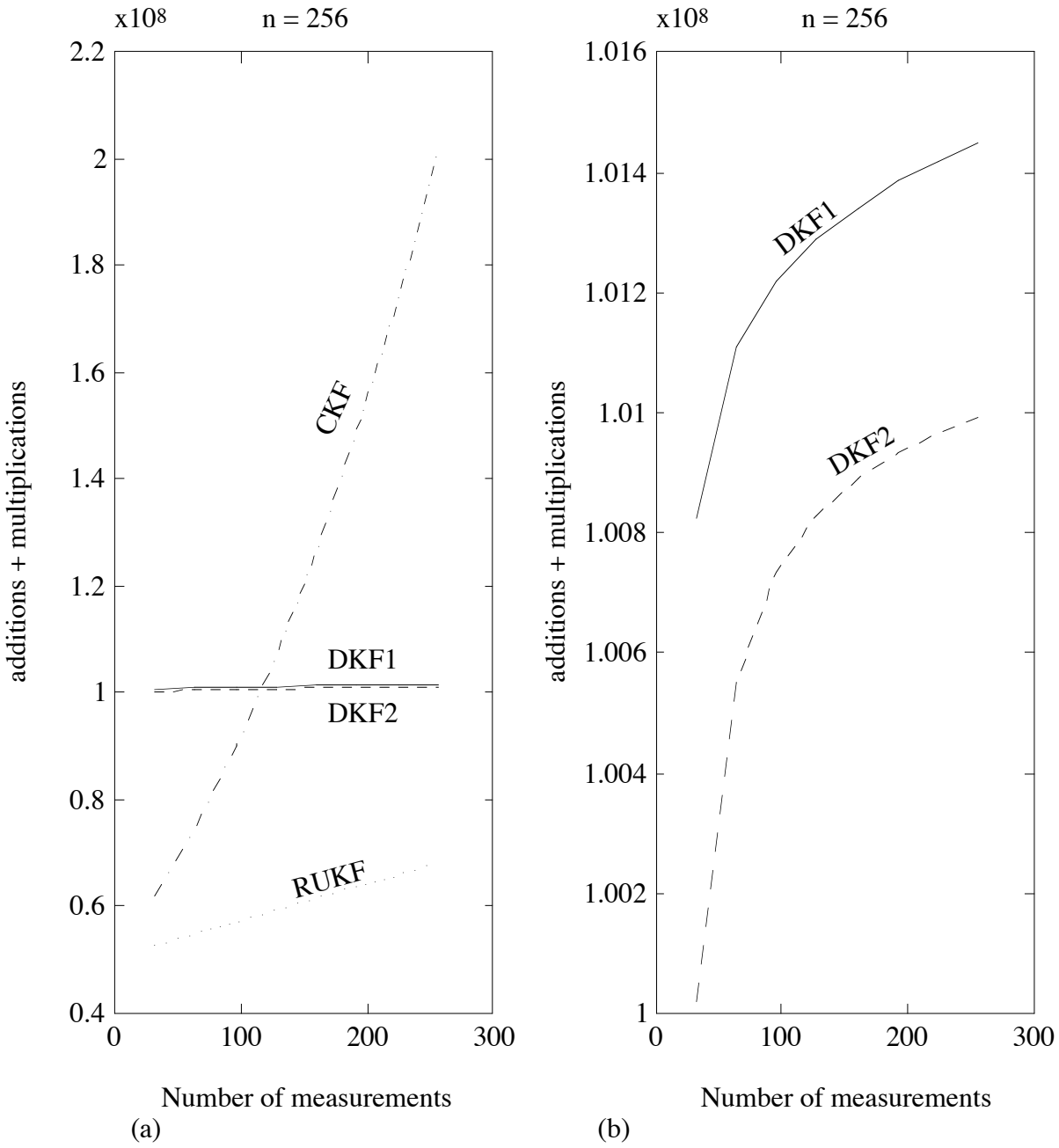

Fig. 3: Evolution of the number of elementary operations in term of the number of outputs when all subsystems have the same dimensions.

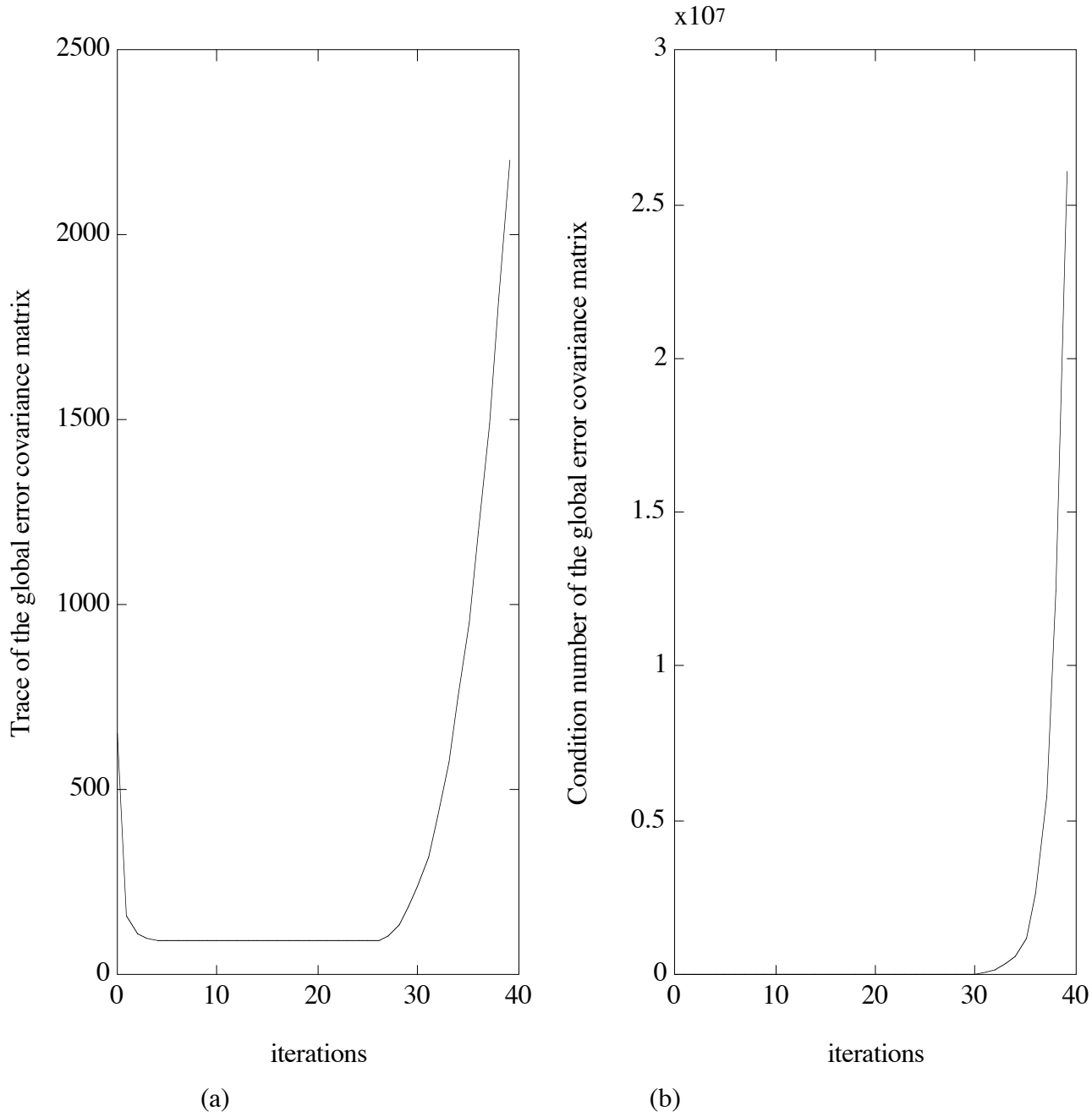

Fig. 4: Effect of round-off errors on the conventional Kalman filter.

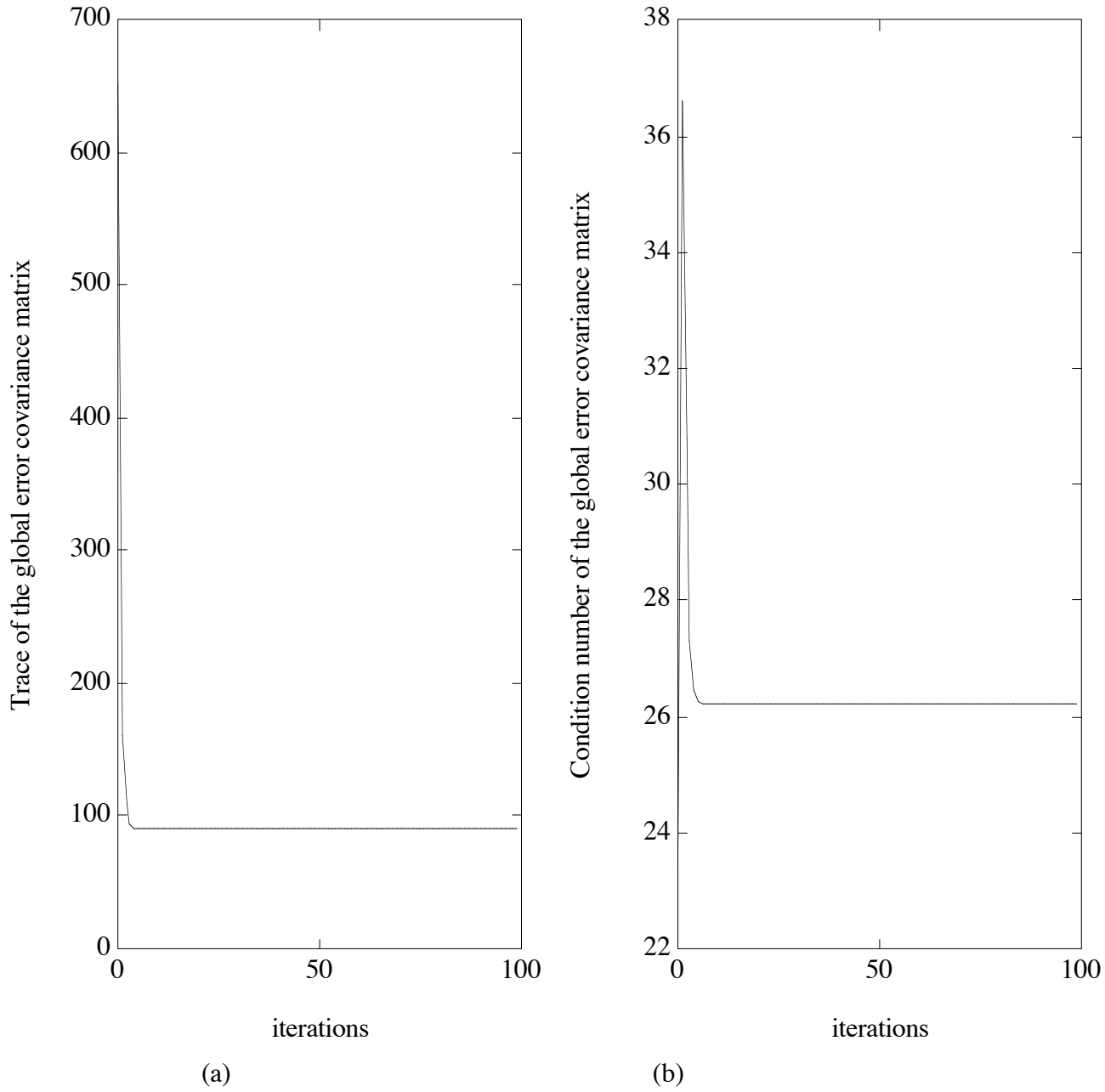

Fig. 5: Effect of round-off errors on the RU Kalman filter and the RU output decentralized filter with a component-wise partitioning of measurements and 10 local processors.

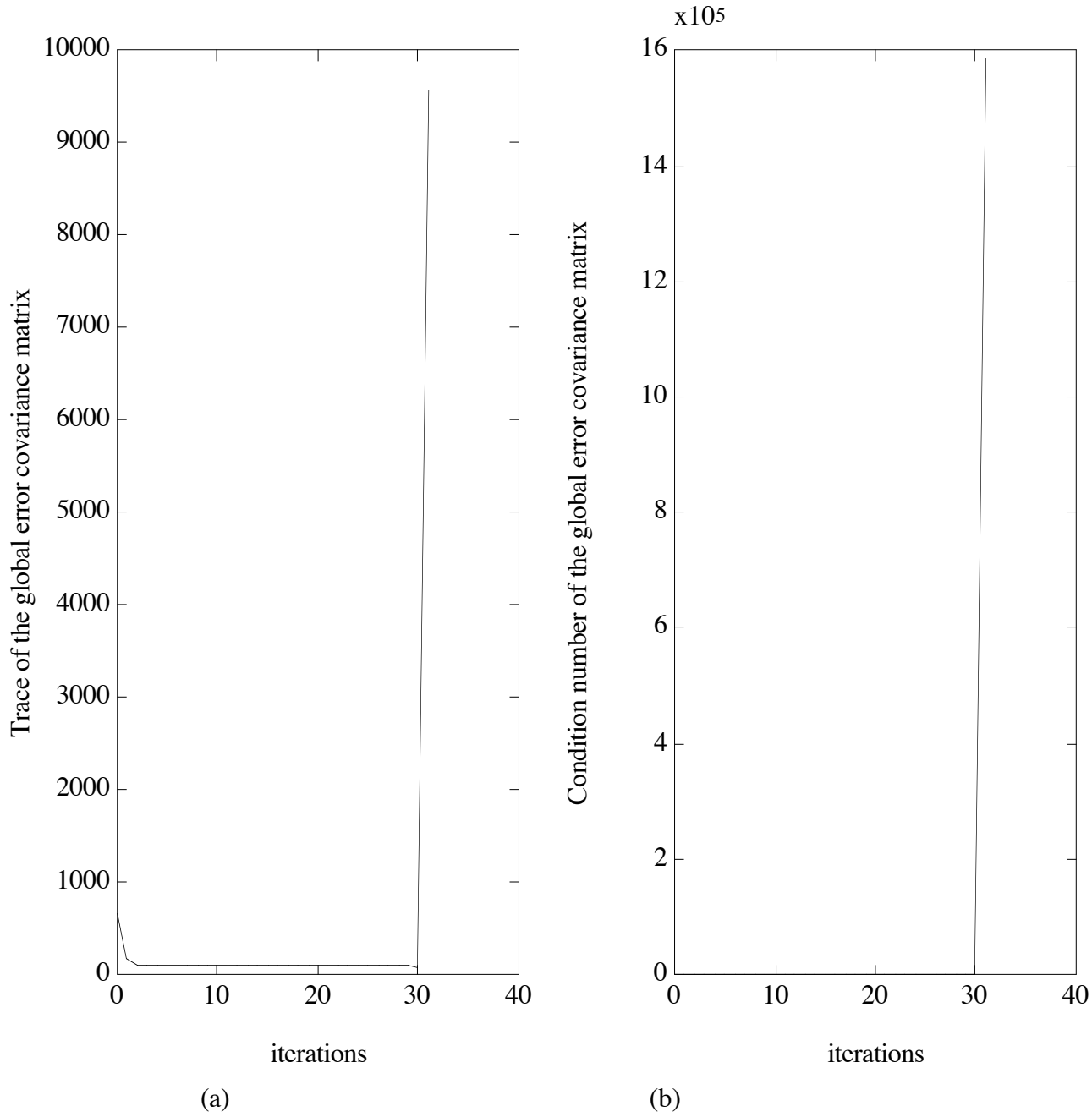

Fig. 6: Effect of round-off errors on the RU Kalman filter when the partitioning of measurements is not a component-wise.

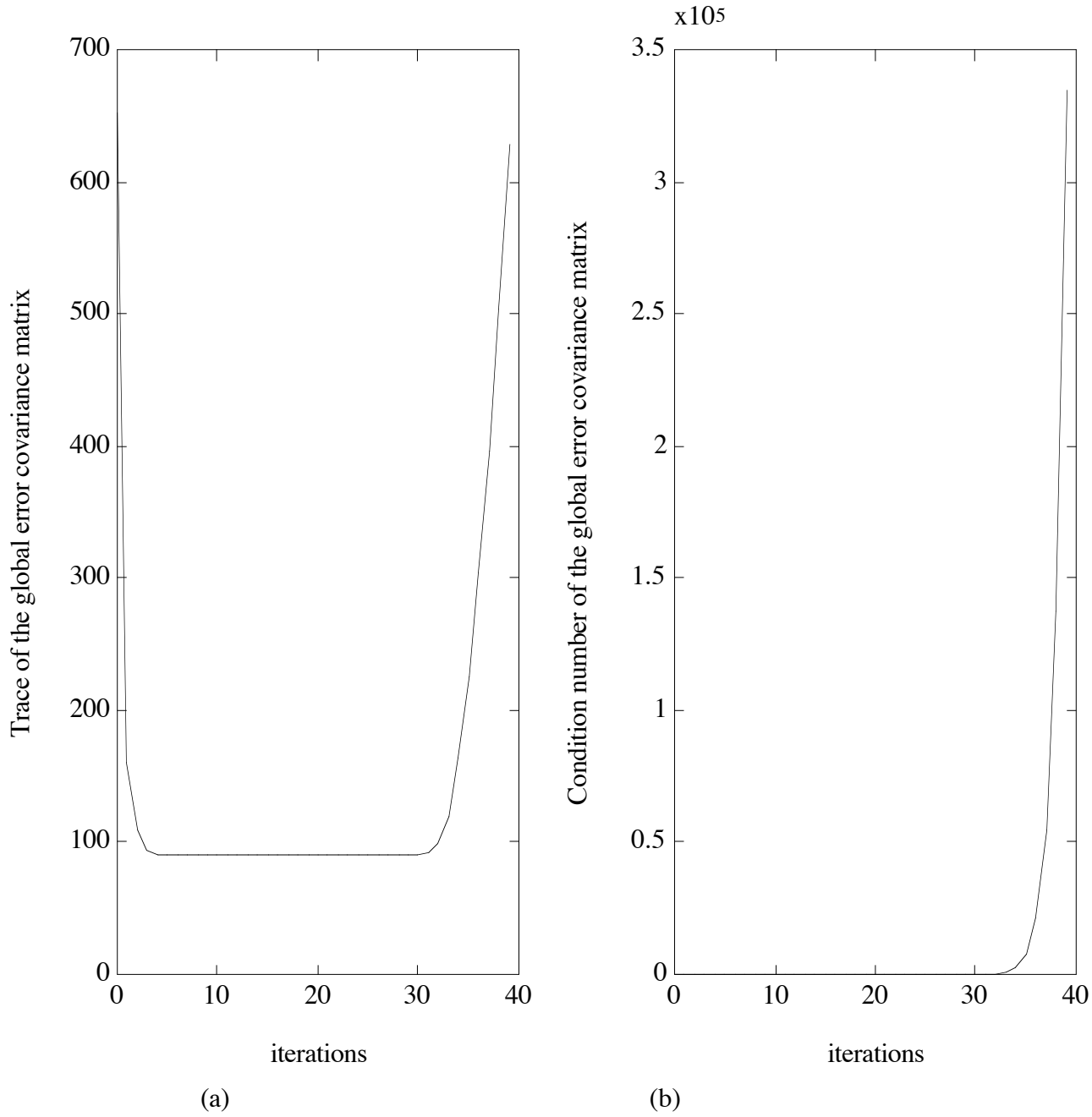

Fig. 7: Effect of round-off errors on Algorithm 4.1 when the local measurements are not scalar.

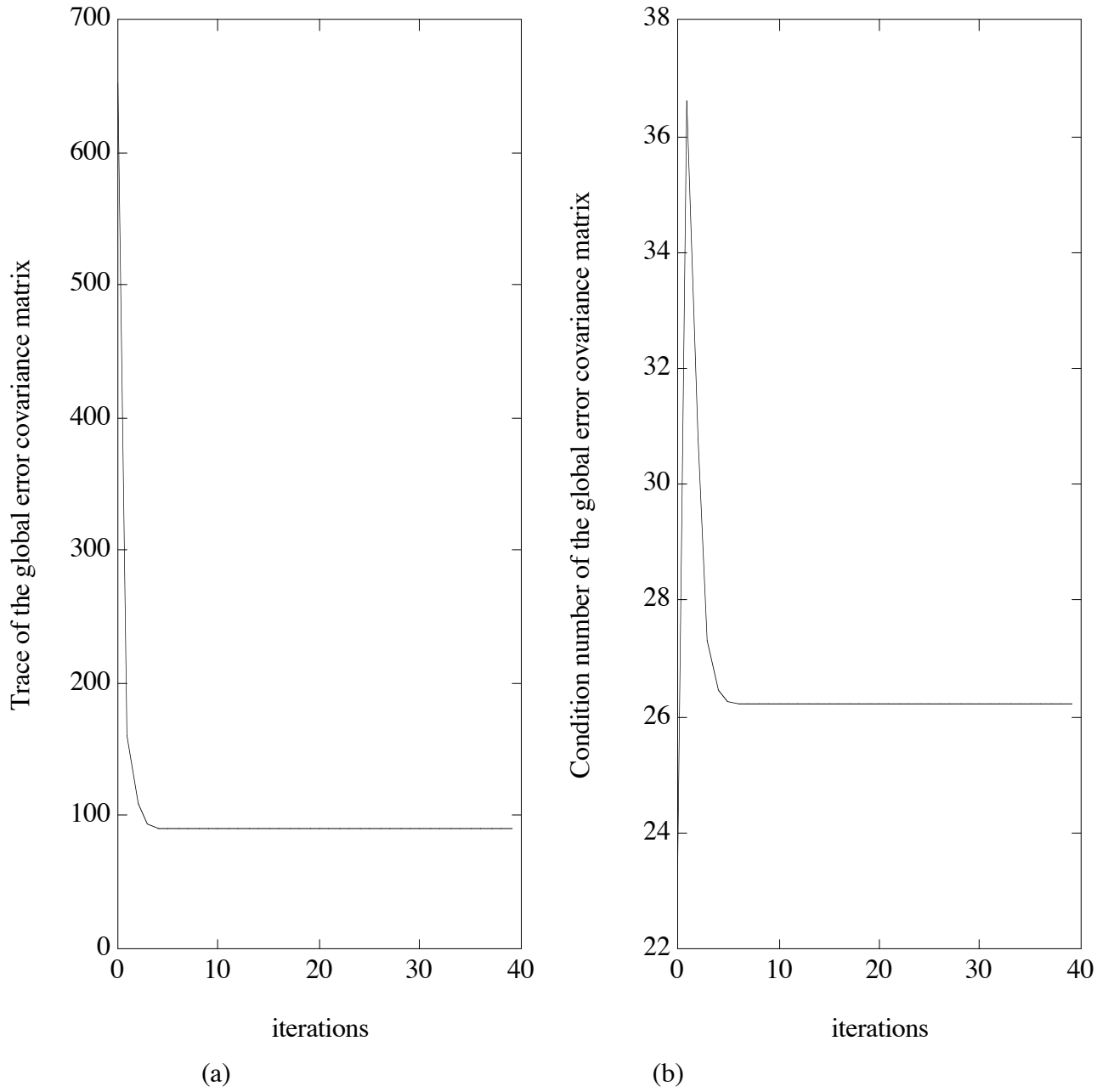

Fig. 8: Effect of round-off errors on Algorithm 4.2 when the local measurements are not scalar.