ON EFFICIENT IMPLEMENTATION OF THE KALMAN FILTER BANK FOR SCENARIO ANALYSIS

Julia V. Tsyganova* and Alexey E. Kondratiev†

*Department of Mathematics and Mechanics, Ulyanovsk State University, 42 L. Tolstoy Str. 432970 Ulyanovsk, Russia
e-mail: TsyganovaJuV@ulsu.ru

†Department of Mathematics and Mechanics, Ulyanovsk State University, 42 L. Tolstoy Str. 432970 Ulyanovsk, Russia
e-mail: mouse.fin@mail.ru

Key words: Linear Stochastic Systems, Kalman Filter, System Identification, Adaptive Filtering, Fault Point Detection, Scenario Analysis.

Abstract. The practical aspects of efficient implementation of the Kalman Filter Bank for scenario analysis are considered. Such implementation have to shorten running time, computational cost, and memory demands, required for computer model of the Kalman Filter Bank.

The first requirement is met by concurrent computing. The second and third requirements are satisfied by substitution the Equivalent Kalman Filter Bank for the Standard Kalman Filter Bank.

The paper contains theoretical and practical results, which can be used for scenario analysis and other problems (fault point detection, system identification, pattern recognition, and others).

The work is partially supported by the Russian Ministry of Education (grants No. T02-03.2-3427 and A03-3.16-409).
1 INTRODUCTION

In many cases dynamical stochastic systems are considered as multi-mode systems. The switching from one mode to another is determined by some scenario. There exists the problem of efficient implementation of the detector of possible system modes switching. The basic requirement on such implementation is to shorten:

- running time,
- computational cost, and
- memory demands.

In the linear case the Standard Kalman Filter Bank (SKFB) can be considered as the model of linear stochastic multi-mode system [1], [2], [3]. However, the practical implementation of the Bank is a very costly algorithm in the case of systems with large dimensionality. The aim of this paper is to investigate the efficient implementation of SKFB computer model. It should satisfy the above mentioned requirements.

The first requirement is met by concurrent computing. The practical computer experiments show the considerable running time shortening.

The second and third requirements are satisfied by substitution the Equivalent Kalman Filter Bank (EKFB) [2] for SKFB. The equivalency of SKFB and EKFB is confirmed in this paper by MatLab and Borland Delphi numerical experiments. The mean residual errors here are less than $10^{-10}$ to $10^{-15}$ for the 5-dimensional system with 16 possible scenarios.

The paper contains theoretical comparison of numbers of arithmetical operations needed for SKFB and EKFB. The obtained areas of efficiency are confirmed by the fact that the number of arithmetical operations for EKFB is 30% less than for SKFB in the case of systems with large dimensionality.

Finally, parameters of linear transformation blocks in EKFB are sparse matrices. This fact allows us to apply sparse matrix packing algorithms for the shortening of memory demands.

All obtained results confirm the efficiency of considered Kalman Filter Bank implementation for scenario analysis and other problems (for example, fault point detection and identification problems).

2 PROBLEM STATEMENT

Let the discrete-time system be characterized by the following equations:

\[ x(t_i) = \Phi x(t_{i-1}) + \Psi u(t_{i-1}) + \Gamma w(t_{i-1}), \]  
\[ z(t_i) = H x(t_i) + v(t_i), \]
where $x(t_i)$ is the $n$-dimensional state vector, $z(t_i)$ is the $m$-dimensional system output, $u(t_i)$ is the control input, and \{w(t_0), w(t_1), \ldots\} and \{v(t_1), v(t_2), \ldots\} are mutually independent zero-mean Gaussian sequences of independent vectors. Without loss of generality, their covariances $Q$ and $R$ are assumed to be reduced to identity matrices: $Q = I$ and $R = I$. This can be easily done by normalizing input noise in (1) and measurements in (2). The sequences are considered independent of Gaussian initial $x(t_0)$ with mean $\bar{x}(t_0)$ and $P(t_0)$.

Let us assume that the system (1), (2) is time invariant, that allow to consider using a steady-state constant-gain Kalman filter model, which is denoted with the subscript $k$.

\begin{align*}
x_k(t_i) &= \Phi_k x_k(t_{i-1}) + \Psi_k u(t_{i-1}) + \Gamma_k w_k(t_{i-1}), \quad (3) \\
z_k(t_i) &= H_k x_k(t_i) + v_k(t_i), \quad (4)
\end{align*}

where $x_k$ is the Kalman filter model state vector, $\Phi_k$ is the Kalman filter model state transition matrix, $\Psi_k$ is the Kalman filter model control input matrix, $u$ is the system input vector, $\Gamma_k$ is the Kalman filter model noise input matrix, $w_k$ is an additive white discrete-time dynamics noise input used in the Kalman filter model, with zero mean and identity covariance, $z_k$ is the Kalman filter model measurement vector, $H_k$ is the Kalman filter model output matrix, $v_k$ is an additive white measurement noise input that is used in the Kalman filter model. It is assumed to be independent of $w_k$, and zero mean with identity covariance.

The Kalman filter algorithm uses this model to define time propagation and measurement update equations of the Kalman filter state estimates and the Kalman filter state estimates covariance matrix. Thus we have

**Time propagation**

\[ \hat{x}_k(t_i^-) = \Phi_k \hat{x}_k(t_{i-1}^-) + \Psi_k u(t_{i-1}), \quad P_k(t_i^-) = \Phi_k P_k(t_{i-1}^-) \Phi_k^T + \Gamma_k \Gamma_k^T. \]

**Vector measurement update**

\begin{align*}
K_k(t_i) &= P_k(t_i^-) H_k^T C_k(t_i)^{-1}, \\
C_k(t_i) &= H_k P_k(t_i^-) H_k^T + I_m, \\
\nu_k(t_i) &= z(t_i) - H_k \hat{x}_k(t_i^-),
\end{align*}

\[ P_k(t_i^+) = (I_n - K_k(t_i)) H_k P_k(t_i^-) (I_n - K_k(t_i)), \]

\[ \hat{x}_k(t_i^+) = \hat{x}_k(t_i^-) + K_k \nu_k(t_i), \]

where $\nu_k(t_i)$ is the Kalman filter residual vector.

The following sequences

\[ N_k = \{ \nu_k(t_1), \nu_k(t_2), \ldots, \nu_k(t_i) \} \]

consists of mutually independent entries $\nu_k(t_\tau) = z_k(t_\tau) - H_k \hat{x}_k(t_\tau^-)$, $\tau = 1, 2, \ldots, t$ with zero mean and the Kalman filter-computed residual covariance matrix $C_k$, subject to the corresponding Kalman filter model. The steady state Kalman filter gain $K_k$ and the steady state Kalman filter residual covariance matrix $C_k$ can be precomputed using (5),
and therefore do not need to be computed in real time. The steady state Kalman filter equations become

\[
\hat{x}_k(t_i^-) = \Phi_k \hat{x}_k(t_{i-1}^-) + \Psi_k u(t_{i-1}),
\]
(7)

\[
\hat{x}_k(t_i^+) = \hat{x}_k(t_i^-) + K_k \nu_k(t_i),
\]
(8)

The set of Kalman filter models forms the Standard Kalman Filter Bank (SKFB).

Further, let us follow [2] to apply expressions for the equivalent residual of a Kalman filter using the residual of another Kalman filter and the known differences between the two Kalman filter models. The subscripts \(j\) and \(k\) denotes the two different Kalman filter models. We consider model differences in the state transition matrix \(\Phi\), the input matrix \(\Psi\), the noise input matrix \(\Gamma\), and the output matrix \(H\).

3 THE STANDARD KALMAN FILTER BANK AND ITS EFFICIENT IMPLEMENTATION: THE EQUIVALENT KALMAN FILTER BANK

Let us use the following notation:

\[
\Delta \Psi_{kj} = \Psi_k - \Psi_j, \quad \Delta \Phi_{kj} = \Phi_k - \Phi_j, \quad \Delta H_{kj} = H_k - H_j, \quad \Delta \Gamma_{kj} = \Gamma_k - \Gamma_j, \quad \Delta K_{kj} = K_k - K_j.
\]

The difference in state estimate errors is defined in the following way:

\[
\Delta \epsilon_{jk}(t_i^+) = \epsilon_j(t_i^+) - \epsilon_k(t_i^+) = [x_T(t_i) - \hat{x}_j(t_i^+)] - [x_T(t_i) - \hat{x}_k(t_i^+)] = \hat{x}_k(t_i^+) - x_j(t_i^+) \rightarrow \hat{x}_j(t_i^+) = \hat{x}_k(t_i^+) - \Delta \epsilon_{jk}(t_i^+).
\]

Let us define the residual of filter number \(j\):

\[
\nu_j(t_i) = z(t_i) - H_j \Phi_j \hat{x}_k(t_{i-1}^-) - H_j \Psi_j u(t_{i-1}) + H_j \Phi_j \Delta \epsilon_{jk}(t_i^+).
\]

This equation can be rewritten in the following way:

\[
\nu_j(t_i) = \nu_k(t_i) + H_j \Phi_j \Delta \epsilon_{jk}(t_{i-1}^-) + [H_j \Delta \Phi_{kj} + \Delta H_{kj} \Phi_k] \hat{x}_k(t_{i-1}^-) + [H_j \Delta \Psi_{kj} + \Delta H_{kj} \Psi_k] u(t_{i-1}).
\]

Let us write a recursive expression for the difference in state estimates between filter \(k\) and filter \(j\) \((\Delta \epsilon_{jk}(t_i^+))\). It allows to compute \(\Delta \epsilon_{jk}(t_i^+))\) at each time sample and then the residual for the filter \(j\). Thus, we have

\[
\Delta \epsilon_{jk}(t_i^+) = (I - K_j H_j) \Phi_j \Delta \epsilon_{jk}(t_{i-1}^-) + [(I - K_k H_k) \Phi_k - (I - K_j H_j) \Phi_j] \hat{x}_k(t_{i-1}^-) + (I - K_k H_k) \Psi_k - (I - K_j H_j) \Psi_j \nu(t_{i-1}) + \Delta K_{kj} z(t_i).
\]

Now consider four cases:
• differences in the input matrix $\Psi$,
• differences in the output matrix $H$,
• differences in the state transition matrix $\Phi$,
• differences in the noise input matrix $\Gamma$.

**Case 1.** $\Psi_k \neq \Psi_j$, $H_k = H_j$, $\Phi_k = \Phi_j$, $\Gamma_k = \Gamma_j$, $K_k = K_j$.

$$\Delta \varepsilon_{jk}(t^+_i) = (I - K_j H_j) \Phi_j \Delta \varepsilon_{jk}(t^+_i) + (I - K_j H_j) \Delta \Psi_{kj} u(t_{i-1}),$$

$$\nu_j(t_i) = \nu_k(t_i) + H_j \Phi_j \Delta \varepsilon_{jk}(t^+_i) + H_j \Delta \Psi_{kj} u(t_{i-1}).$$

**Case 2.** $\Psi_k = \Psi_j$, $H_k \neq H_j$, $\Phi_k = \Phi_j$, $\Gamma_k = \Gamma_j$, $K_k \neq K_j$.

$$\Delta \varepsilon_{jk}(t^+_i) = (I - K_j H_j) \Phi_j \Delta \varepsilon_{jk}(t^+_i) + \Delta K_{kj} z(t_i) + (K_j H_j - K_k H_k) \tilde{\hat{x}}_k(t^-_i),$$

$$\nu_j(t_i) = \nu_k(t_i) + H_j \Phi_j \Delta \varepsilon_{jk}(t^+_i) + \Delta H_{kj} \tilde{\hat{x}}_k(t^-_i).$$

**Case 3.** $\Psi_k = \Psi_j$, $H_k = H_j$, $\Phi_k \neq \Phi_j$, $\Gamma_k = \Gamma_j$, $K_k \neq K_j$.

$$\Delta \varepsilon_{jk}(t^+_i) = (I - K_j H_j)[\Phi_j \Delta \varepsilon_{jk}(t^+_i) + \Phi_k \tilde{\hat{x}}_k(t^-_i)] + \Delta K_{kj} \nu_k(t_i),$$

$$\nu_j(t_i) = \nu_k(t_i) + H_j \Phi_j \Delta \varepsilon_{jk}(t^+_i) + H_j \Delta \Phi_{kj} \tilde{\hat{x}}_k(t^-_i).$$

**Case 4.** $\Psi_k \neq \Psi_j$, $H_k = H_j$, $\Phi_k = \Phi_j$, $\Gamma_k \neq \Gamma_j$, $K_k \neq K_j$.

$$\Delta \varepsilon_{jk}(t^+_i) = (I - K_j H_j) \Phi_j \Delta \varepsilon_{jk}(t^+_i) + \Delta K_{kj} z(t_i),$$

$$\nu_j(t_i) = \nu_k(t_i) + H_j \Phi_j \Delta \varepsilon_{jk}(t^+_i).$$

So, the Equivalent Kalman Filter Bank (EKFB) consists of the source Kalman filter $k$ and the set of linear transforms for producing the equivalent of the residual from another Kalman filters $j$ ($j = 1, \ldots, M$).

## 4 COMPARISON OF THE OPERATION COUNT. AREAS OF EFFICIENT IMPLEMENTATION OF EKFB

Let us compare the operation count (number of multiplications and additions) that are needed to implement the Standard Kalman Filter Bank with the count needed to implement the Equivalent Kalman Filter Bank. Consider the following values: $n_s$ — dimension of state vector $x$, $n_i$ — dimension of control input $u$, $n_m$ — dimension of measurement vector $z$.

Let us denote the number of multiplications as $< * >$ and the number of additions as $< + >$.

<table>
<thead>
<tr>
<th>SKFB</th>
<th>EKFB</th>
</tr>
</thead>
<tbody>
<tr>
<td>$&lt; * &gt;$</td>
<td>$n_s(n_s + n_i + 2n_m)$</td>
</tr>
<tr>
<td>$&lt; + &gt;$</td>
<td>$n_s(n_s + n_i + 2n_m - 1)$</td>
</tr>
</tbody>
</table>
EKFB, $\Delta \Psi \neq 0$.
Let $n_\Psi$ — the number of nonzero columns of matrix $\Delta \Psi$ ($1 \leq n_\Psi \leq n_i$). Then

\[
<\ast> = n_\Psi (n_s + n_m) + n_s (n_s + n_m), \\
<+> = n_\Psi (n_s + n_m) + n_s (n_s + n_m - 2).
\]

Let us define the area of efficient implementation of the EKFB through examining the set of inequalities which describes the gain in the operation count.

Choosing from (9) the stronger inequality, we have:

\[
n_\Psi \leq n_s (n_i + n_m) / (n_s + n_m).
\]

Let’s notice, that this inequality is meaningful always, since $n_s$ and $n_m$ are positive natural numbers.

Taking into account the possible values of $n_\Psi$, we can draw a plot of the desired area (see figure 1). In this case efficiency is particularly obvious for systems with $(n_i << n_s)$ & $(n_i << n_m)$.

![Figure 1: Area of efficiency in Case 1.](image-url)

**EKFB, $\Delta H \neq 0$ ($\rightarrow \Delta K \neq 0$).**
Let $n_H$ be the number of nonzero rows of matrix $\Delta H$ ($1 \leq n_H \leq n_m$), and $n_K$ the size of nonzero block of matrix $\Delta K$ ($1 \leq n_K \leq \min\{n_s, n_m\}$). Then

$$< \ast > = n_H(2n_s - n_K) + n_K(n_s + n_K) + n_s(n_s + n_m),$$

$$< + > = n_H(2n_s - n_K - 1) + n_K(n_s + n_K - 1) + n_s(n_s + n_m).$$

There are two possible alternatives in this case:

1) If $n_H \geq n_s - n_K$, then

$$n_H \leq (n_s(n_i + n_m) - n_K(n_s + n_K))/(2n_s - n_K); \quad (11)$$

2) If $n_H < n_s - n_K$, then

$$n_H \leq (n_s(n_i + n_m - 1) - n_K(n_s + n_K - 1))/(2n_s - n_K - 1). \quad (12)$$

Let us consider these alternatives in order:

1) The inequality (11) is meaningful always, because $n_k \leq n_s$, that is $2n_s - n_K \neq 0$. Taking into account restrictions on $n_H$, it is possible to draw a plot of desired area (see figure 2). In this case efficiency grows at increase $n_i$.

2) The denominator of fraction in the right part of an inequality (12) can be equal zero in case $n_K = 2n_s - 1$, i.e. $n_K - n_s = n_s - 1$, but since $n_K - n_s \leq 0$, and $n_s - 1 \leq 0$. It is possible only at $n_s = 1$. But then $n_s = 1, n_K = 1$ and $1 \leq 0$, since $n_H \leq n_s - n_K$. We come to contradiction. Hence, a denominator of fraction is not equal to zero at any allowable values of parameters, therefore the inequality (12) is meaningful always. Now we can draw a plot of desired area subject to restrictions on values $n_H$ (see figures 3). In this case efficiency quickly decreases at increase of system dimensions.

**EKFB, $\Delta \Phi \neq 0 \rightarrow \Delta K \neq 0$.**

$$< \ast > = n_s(2n_s + 3n_m),$$

$$< + > = n_s(2n_s + 3n_m - 1) - 2n_m.$$
The computational savings are obvious for systems with \((n_i >> n_s)\) \& \((n_i >> n_m)\).

\[ \Delta \Gamma \neq 0 \quad (\rightarrow \Delta K \neq 0). \]

\[ < * > = n_s(n_s + 2n_m), \]
\[ < + > = n_s(n_s + 2n_m - 1) - n_m. \]

Here the sparse structure of matrix \(\Delta \Gamma\) does not give advantages too, however computational savings are evident based on inequalities \(n_i \geq 0\) and \(n_i n_s + n_m \geq 0\).

All obtained results allow at comparison EKFB with SKFB for the systems of large dimensionality to speak about considerable computational savings during EKFB implementation.

5 NUMERICAL EXAMPLE

As a practical numerical example consider the following system taken from the inertial navigation [4]:

\[ n_i n_s(n_i + n_m) - n_s(n_s + n_m)(2n_s - n_K) \]
\[ n H = n + m - n K \]
\[ N H = (n s (n m + n i - 1) - n k (n s + n k - 1)) / (2n s - n k - 1) \]

Figure 3: Area of efficiency in Case 2.2

\[
x_{t+1} = \begin{bmatrix} 0.75 & -1.74 & -0.3 & 0.0 & -0.15 \\
0.09 & 0.91 & -0.0005 & 0.0 & -0.008 \\
0.0 & 0.0 & 0.95 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.55 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.905 \end{bmatrix} x_t + \begin{bmatrix} 0.0 \\
0.0 \\
0.0 \\
0.0 \\
0.0 \end{bmatrix} + w_t
\]

\[
z_t = \begin{bmatrix} 1 - e & 0 & 0 & 0 & 1 - f \\
0 & 1 - g & 0 & 1 - h & 0 \end{bmatrix} x_t + v_t, \quad e, f, g, h = \{0, 1\}
\]

\{w_t\} and \{v_t\} are zero-mean white Gaussian sequences with covariances \(Q_t = I_3\), and \(R_t = I_2\) (\(I_n\) is the \(n\)-dimensional identity matrix). These equations describe the damped Shuler loop driven by the exponentially correlated 3-dimensional noise \(w_t\) [4].

The mode of the stochastic system, or system scenario, is determined by matrix \(H\). Thus, \(2^4 = 16\) types of system scenario are possible.

Table 1 shows experimental data, which confirm the equivalence of residuals obtained from the Standard Kalman Filter Bank and the Equivalent Kalman Filter Bank. This experiment was conducted for the chosen example and 1000 quantums of time. The source filter corresponds to scenario \(S_0\). The mean residual errors here are less than \(10^{-14}\) for
the 5-dimensional system with 16 possible scenarios.

6 SCENARIO ANALYSIS

Suppose that there is a control interval \([t_0, t_N]\) on which the system behavior is subject to some scenario \(S_0\). It means that system parameters are \(\Phi_0\), \(\Psi_0\), \(\Gamma_0\), and \(H_0\).

Further, consider the time moment \(t_s \in [t_0, t_N]\) at which the system behavior can be changed, i.e., the nominal scenario \(S_0\) is changed to one of possible scenarios \(S_k\), \(k = 1 \ldots , M\). Each scenario \(S_k\) is considered as the set of the system parameters \(\{\Phi_k, \Psi_k, \Gamma_k, \text{ and } H_k\}\).

Our goal is to choose the right scenario of system behavior after the moment \(t_s\). One of the possible solving method is the Wald sequential probability ratio test [5], which requires the computation of likelihood ratio function \(\lambda_p(t_i)\) at each moment \(t_i\). Thus, we have

\[
\lambda_p(t_i) = \lambda_p(t_{i-1}) + \mu_p(t_i),
\]

\[
2\mu_p(t_i) = \ln \det C_0(t_i) - \ln \det C_p(t_i) + (s_0(t_i) - s_p(t_i)),
\]
Difference of Residuals

<table>
<thead>
<tr>
<th>Type of Scenario</th>
<th>Difference of Residuals $\nu_1$</th>
<th>Difference of Residuals $\nu_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_0$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$S_1$</td>
<td>$4.64 \cdot 10^{-14}$</td>
<td>$1.91 \cdot 10^{-14}$</td>
</tr>
<tr>
<td>$S_2$</td>
<td>$4.79 \cdot 10^{-14}$</td>
<td>$6.02 \cdot 10^{-15}$</td>
</tr>
<tr>
<td>$S_3$</td>
<td>$4.79 \cdot 10^{-14}$</td>
<td>$5.12 \cdot 10^{-16}$</td>
</tr>
<tr>
<td>$S_4$</td>
<td>$4.31 \cdot 10^{-14}$</td>
<td>$2.70 \cdot 10^{-14}$</td>
</tr>
<tr>
<td>$S_5$</td>
<td>$4.37 \cdot 10^{-14}$</td>
<td>$2.65 \cdot 10^{-14}$</td>
</tr>
<tr>
<td>$S_6$</td>
<td>$4.32 \cdot 10^{-14}$</td>
<td>$6.02 \cdot 10^{-15}$</td>
</tr>
<tr>
<td>$S_7$</td>
<td>$4.32 \cdot 10^{-14}$</td>
<td>$5.12 \cdot 10^{-16}$</td>
</tr>
<tr>
<td>$S_8$</td>
<td>$1.45 \cdot 10^{-14}$</td>
<td>$2.18 \cdot 10^{-14}$</td>
</tr>
<tr>
<td>$S_9$</td>
<td>$1.37 \cdot 10^{-14}$</td>
<td>$2.39 \cdot 10^{-14}$</td>
</tr>
<tr>
<td>$S_{10}$</td>
<td>$1.33 \cdot 10^{-14}$</td>
<td>$6.02 \cdot 10^{-15}$</td>
</tr>
<tr>
<td>$S_{11}$</td>
<td>$1.33 \cdot 10^{-14}$</td>
<td>$5.12 \cdot 10^{-16}$</td>
</tr>
<tr>
<td>$S_{12}$</td>
<td>$3.12 \cdot 10^{-15}$</td>
<td>$2.21 \cdot 10^{-14}$</td>
</tr>
<tr>
<td>$S_{13}$</td>
<td>$3.12 \cdot 10^{-15}$</td>
<td>$2.42 \cdot 10^{-14}$</td>
</tr>
<tr>
<td>$S_{14}$</td>
<td>$3.12 \cdot 10^{-15}$</td>
<td>$6.02 \cdot 10^{-15}$</td>
</tr>
<tr>
<td>$S_{15}$</td>
<td>$3.12 \cdot 10^{-15}$</td>
<td>$5.12 \cdot 10^{-16}$</td>
</tr>
<tr>
<td><strong>256 ensemble average</strong></td>
<td><strong>$4.59 \cdot 10^{-14}$</strong></td>
<td><strong>$1.78 \cdot 10^{-14}$</strong></td>
</tr>
</tbody>
</table>

Table 1: Average Difference between SKFB and EKFB residuals.

\[
s_0(t_i) = \nu_0^T(t_i)C_0^{-1}(t_i)\nu_0(t_i),
\]
\[
s_p(t_i) = \nu_p^T(t_i)C_p^{-1}(t_i)\nu_p(t_i),
\]

where $\nu_0(t_i)$ is a system residual for scenario $S_0$, $\nu_p(t_i)$ is a system residual for scenario $S_p$ ($1 \leq p \leq M$), and matrices $C_0(t_i)$ and $C_p(t_i)$ have to be precomputed up to steady state values $C_0$ and $C_p$.

The value of $\lambda_p(t_i)$ is then tested against two threshold levels $A$ and $B$ (where $A > B$). If a priori probabilities of different types of scenario are equal, then initial value of $\lambda_p(t_0) = 0$.

Decision rule for choosing desired scenario is as follows [6]:

1. If $\forall p \lambda_p(t_i) \leq B$, then system scenario $S_0$ is chosen.
2. If $\exists q \forall p \neq q (\lambda_q(t_i) \geq A) \& (\lambda_p(t_i) < A)$, then system scenario $S_q$ is chosen.
3. If $\forall p B < \lambda_p(t_i) \leq A$, the process is repeated for $t_{i+1}$.
4. If $\exists p, q \ (p \neq q): (\lambda_p(t_i) \geq A) \& (\lambda_q(t_i) \geq A)$, then the process is terminated with the choice of scenario $S_i$, for which the value of likelihood function is maximum, that is $\lambda_l(t_i) = \max \{\lambda_p(t_i), \lambda_q(t_i)\}$. (13)
The probability of the right choice of scenario varies within the limits of 60%-95%, that directly depends on parameters of a concrete system.

7 CONCURRENT COMPUTING

To satisfy the first requirement mentioned in Section 1, the concurrent computing was implemented for SKFB and EKFB computer models.

A shortening of running time within the framework of the given experiment to estimate difficultly valid its transiency, however on 10000 passes of corresponding testing intervals the following parameters of efficiency of EKFB have been obtained in comparison with SKFB (The testing was carried out on CPU Intel Pentium II Original, 64 Mb RAM, OS Microsoft Windows 98 OSR2, the source code was compiled in Borland Delphi 5 Enterprize Edition Version 5.0 Build 5.62.):

- EKFB (concurrent programming did not applied) \(\approx 11\%\);
- EKFB (concurrent programming was applied, thread priority is \(tpNormal\)) \(\approx 68.8\%\);
- EKFB (concurrent programming was applied, thread priority is \(tpHigher\)) \(\approx 66.2\%\);
- EKFB (concurrent programming was applied, thread priority is \(tpHighest\)) \(\approx 65.03\%\);

The given proposal is probably effective for problems with large, growing in time volumes of complex calculations.

8 IMPLEMENTATION OF SPARSE MATRICES

Distinctions between models in the bank of filters are often represented by sparse matrices, therefore it is reasonable to apply packing algorithms for their storage and processing. The next way of packing has been applied for realization of experiments (an initial code in Borland Delphi 5):

```delphi
TFloat = Extended;
TMatrixBaseClass = class
end;
TInfoRec = record
  i, j: Integer;
  Data: TFloat;
end;
TPackedMatrixArray = array of TInfoRec;
TPackedMatrix = class(TMatrixBaseClass)
private
  Fi: Integer;
  Fj: Integer;
end;
```
The packed matrix is submitted as an array (TPackedMatrixArray) of open length which elements are information records (TInfoRec). Elements of a matrix are numbered in the lines from the first column to the last. Such implementation gives serious economy of operative memory for a data storage and gives an opportunity rather easy in sense of amount of operations of search of a necessary element of the packed matrix.

Here it is necessary to mention accuracy of storage of matrix elements. Insufficient accuracy can lead to the situation in which at definition of distinctions between system models required sparse matrices will turn out zero, that certainly will lead to occurrence of incorrect results.

9 CONCLUSIONS

The purpose of the given work was theoretical research, development of idea [2], [3] and computer implementation of efficient bank of Kalman filters, experimental confirmation of its equivalence to the Standard Kalman Filter Bank, and comparison of operation count. The possibility of using the Equivalent Kalman Filter Bank in scenario analysis with application of concurrent programming and Win32 threads technology was also considered.

There was obtained the following theoretical results: the idea of EKFB was developed, theoretical number of arithmetical operations for EKFB in comparison this required number of operations for SKFB was obtained in the form of inequalities and graphically as the areas of efficiency.

The practical results are the following: the efficient (equivalent) bank of Kalman filters with application of concurrent computing and packing algorithms of sparse matrices was implemented, its equivalence to SKFB was experimentally proved.

Thus, considered implementation of the Kalman Filter Bank is really efficient, because it meets the three basic requirements: shortening of running time, computational cost, and memory demands. This implementation can be used for scenario analysis, fault point detection, system identification, pattern recognition, and other problems.
REFERENCES


