Eigenfactor Solution of the Matrix Riccati Equation—A Continuous Square Root Algorithm

YAAKOV OSHMAN AND ITZHACK Y. BAR-ITZHACK, SENIOR MEMBER, IEEE

Abstract—**This paper introduces a new algorithm for solving the matrix Riccati equation. Differential equations for the eigenvalues and eigenvectors of the solution matrix are developed in which their derivatives are expressed in terms of the eigenvalues and eigenvectors themselves and not as functions of the solution matrix. The solution of these equations yields, then, the time behavior of the eigenvalues and eigenvectors of the solution matrix. A reconstruction of the matrix itself at any desired time is immediately obtained through a trivial similarity transformation.**

This algorithm serves two purposes. First, being a square root solution, it entails all the advantages of square root algorithms such as nonnegative definiteness and accuracy. Secondly, it furnishes the eigenvalues and eigenvectors of the solution matrix continuously without resorting to the complicated route of solving the equation directly and then decomposing the solution matrix into its eigenvalues and eigenvectors.

The algorithm which handles cases of distinct as well as multiple eigenvalues is tested on several examples. Through these examples it is seen that the algorithm is indeed more accurate than the ordinary one. Moreover, it is seen that the algorithm works in cases where the ordinary algorithm fails and even in cases where the closed-form solution cannot be computed as a result of numerical difficulties.

I. INTRODUCTION

THE matrix Riccati equation

$$\dot{P}(t) = F(t)P(t) + P(t)F(t)^T + Q(t) - P(t)C(t)P(t)$$
$$P(t_o) = P_o,\ t \geq t_o \tag{1.1}$$

in which Q, C, and P_o are $n \times n$ real symmetric matrices and F is an $n \times n$ real matrix, plays, as is well known, a central role in optimal control and estimation [1]–[5]. Suppose that one is interested in a continuous observation of the eigenvalues and eigenvectors (eigencomponents) of the solution $P(t)$. This can be done by a frequent decomposition of $P(t)$ into its eigencomponents. However, due to computational limitations, this approach is inefficient and may even be impossible. It is, therefore, desired to find a set of differential equations whose solution yields the eigencomponents of $P(t)$ directly without a need to decompose $P(t)$ or use it in any other way.

Another interesting aspect to such a solution, if found, is entailed in its being actually a square root solution [6]–[9] to the Riccati equation. This can be seen as follows. Let $V(t)$ be a matrix whose columns are the eigenvectors of $P(t)$ and let $\Lambda(t)^{1/2}$ be a diagonal matrix whose elements are the positive square roots of the eigenvalues of $P(t)$ (they are of course real and nonnegative) then a matrix $W(t)$ can be defined as follows:

$$W(t) \triangleq V(t)\Lambda(t)^{1/2} \tag{1.2}$$

where $W(t)$ is a square root of $P(t)$ since

$$P(t) = W(t)W(t)^T. \tag{1.3}$$

Manuscript received March 16, 1984; revised March 11, 1985. Paper recommended by Associate Editor, T. L. Johnson.

The authors are with the Department of Aeronautical Engineering, Technion-Israel Institute of Technology, Haifa, Israel.

The fact that the eigenfactors of $P(t)$ are solved directly implies that $V(t)$ and $\Lambda(t)^{1/2}$ are obtained directly without resorting to the computation of $P(t)$ itself.

We note here the analogy between this method for solving indirectly the Riccati differential equation, and square root methods that are well known and widely used in discrete filtering problems. Square root methods are used to overcome the inherent numerical instability which is involved in the direct implementation of the discrete measurement Kalman filter, and to increase the accuracy which can be achieved with a given finite wordlength [8], [9]. Of all these methods, in which the estimation error covariance is replaced by its factors, one of the most popular is Bierman's U–D covariance factorization. This algorithm uses a UDU^T factorization of the covariance matrix, and in place of the covariance matrix, its U–D factors are updated after each measurement and are propagated in time. Bierman's method guarantees nonnegativity of the computed covariance, and is numerically stable and far more accurate than the original covariance filter. Cases that are strongly ill-conditioned may also arise in the continuous version of the Kalman filter, as is shown later. In these cases, square root methods for solving the Riccati equation are of great value. However, in comparison with discrete-time algorithms, continuous-time algorithms have attracted less interest; this is probably so because in most applications the discrete-time algorithms are more in tune with digital implementations. Several algorithms were developed for the continuous propagation in time of the error covariance square root factors (i.e., for the solution of Lyapunov's equation). Andrews [10] and Tapley and Choe [11] proposed algorithms for the triangular square roots of $P(t)$. Tapley and Peters [12] proposed an algorithm for the time propagation of the U–D factors of $P(t)$. All of these algorithms have to be combined with some discrete square root measurement update algorithm, in order to yield a complete continuous-discrete square root estimator. Morf, Levy, and Kailath [13] addressed the continuous-time estimation problem and presented a square root algorithm for the solution of the nonlinear covariance (Riccati) equation. Their algorithm uses a lower triangular square root factor of $P(t)$ and resembles the algorithms of Andrews, and Tapley and Choe, from a computational standpoint.

In this paper we present a method by which, given the matrix Riccati differential equation, one may solve directly for the eigenvalues and eigenvectors of $P(t)$. We further show that this method can be used as a continuous square root algorithm for the solution of the matrix Riccati differential equation. Unlike the discrete filtering case where two different algorithms are needed; namely, one for time propagation and one for measurement update, here, in the continuous case, only one algorithm is needed. We note here that the main difference between the new algorithm and the above-mentioned continuous-time algorithms is the particular choice of the eigenfactors of $P(t)$; namely, $V - \Lambda$ as the "square root" factors to be propagated. There are cases where the differential equations for the propagation of the factors of $P(t)$ in triangular square root algorithms fail because of improper integration implementation. That is, the integration algorithms do not adapt to the singularities associated with "bad geometry" and low measurement noise. The algorithm presented

0018-9286/85/1000-0971$01.00

in the ensuing should not have this problem since, as will be shown later, the algorithm contains logic to identify singularities and eigenvalue cluster.

In the section that follows we derive the differential equations for the eigenvalues and eigenvectors of $P(t)$. We then show how these equations are used as a square root solution of the Riccati equation in the cases of distinct or equal eigenvalues.

Next, the case of close eigenvalues is dealt with, and it is shown that a solution can still be obtained in this case at a price of some added complexity. We demonstrate the analytical results by digital simulations, and show the superior numerical stability of the method. We conclude with a discussion of the results and of other applications of the method.

II. Differential Equations for the Eigencomponents of $P(t)$

A. Rate of Change of the Eigencomponents of Self-Adjoint Matrices

The ensuing development is based on the following theorem.

Theorem 1: Let $P(t)$ be an $n \times n$ complex matrix function which depends on the real parameter t, and let $P(t)$ satisfy the following two conditions for every $t \in R$.

a) $P(t)$ is self adjoint, i.e., $P(t) = P(t)^*$, where the asterisk denotes the conjugate transpose matrix.

b) $P(t)$ is an analytic function of the real variable t.

Then there exist scalar functions $\{\lambda_i(t)\}_{i=1}^{n}$ and a matrix-valued function $V(t)$, which are analytic for $t \in R$ and possess the following properties for every $t \in R$:

i) $P(t) = V(t)^{-1}\ \text{diag}\ [\lambda_1(t), \lambda_2(t), \cdots, \lambda_n(t)]V(t)$;

ii) $V(t)V(t)^* = I$.

For proof of this theorem, see [14].

Note that $P(t)$, the solution of (1.1) satisfies the two conditions of Theorem 1, thus $P(t)$ can be decomposed as follows:

$$P(t) = V(t)\Lambda(t)V(t)^T \tag{2.1}$$

for all $t \in R$ and moreover, $V(t)$ and $\Lambda(t)$ are analytic.

Let us denote the number of different values that the eigenvalues of $P(t)$ take by k and the multiplicity of the ith eigenvalue by m_i. We then denote each of the n eigenvalues of $P(t)$ by ${}^j\lambda_i$ where the superscript j is used to distinguish between the m_i eigenvalues which take the same value λ_i. Obviously, $1 \le j \le m_i$ and $\Sigma_{i=1}^{k} m_i = n$. In order to simplify notations we will use the superscript of the eigenvalues only when necessary. In the ensuing we will make use of the following well-known characteristics of the self-adjoint P matrix [15] (for convenience, from now on we drop the argument t).

a) For the ith eigenvalue λ_i of multiplicity m_i, the dimension of the null space of $[P - \lambda_i I]$, denoted by $N(P - \lambda_i I)$, is m_i.

b) There are m_i linearly independent eigenvectors of P corresponding to λ_i which span the subspace $N(P - \lambda_i I)$. These eigenvectors are orthogonal to all the eigenvectors belonging to λ_q for which $q \neq i$. In analogy to the notation used for the eigenvalues, we denote the eigenvectors belonging to the ith eigenvalue by $\{{}^j v_i\}_{j=1}^{m_i}$.

c) In all, there is a complete set of n linearly independent eigenvectors which span R^n. These vectors can be chosen to be an orthonormal set, i.e.,

$$v_i^T \cdot v_q = \delta_{iq} \tag{2.2}$$

where δ_{iq} is Kronecker's delta function.

d) The n-dimensional Euclidean space R^n is a direct sum of the k eigenspaces corresponding to the k distinct eigenvalues of P; that is,

$$\sum_{i=1}^{k} \oplus N(P - \lambda_i I) = R^n. \tag{2.3}$$

With all this on hand, we now proceed by writing that for each eigenvalue, we have

$$[P - {}^j\lambda_i I]{}^j v_i = 0 \qquad j = 1, \cdots, m_i \qquad i = 1, \cdots, k. \tag{2.4}$$

In the following development we will deal specifically with the ith eigenvalue and its corresponding jth eigenvector ${}^j v_i$, but it is obvious that the results are equally applicable to all eigenvalues and eigenvectors. By virtue of Theorem 1 we may now differentiate (2.4) with respect to time to obtain

$$[P - {}^j\lambda_i I]{}^j\dot{v}_i + [\dot{P} - {}^j\dot{\lambda}_i I]{}^j v_i = 0. \tag{2.5}$$

Since ${}^j\dot{v}_i \in R^n$, it can be expressed as a linear combination of vectors in the null-spaces $N(P - \lambda_q I)$ of which R^n consists; that is,

$${}^j\dot{v}_i = \sum_{q=1}^{k} \sum_{p=1}^{m_q} {}^p\alpha_q {}^p v_q \tag{2.6}$$

where ${}^p v_q \in N(P - \lambda_q I)$.

Equation (2.6) can be rewritten as

$${}^j\dot{v}_i = \sum_{p=1}^{m_i} {}^p\alpha_i {}^p v_i + \sum_{\substack{q=1 \\ q\neq i}}^{k} \sum_{p=1}^{m_q} {}^p\alpha_q {}^p v_q \tag{2.7}$$

where a distinction has been made between the eigenvectors ${}^j v_i \in N(P - \lambda_i I)$ and all the eigenvectors which correspond to eigenvalues other than λ_i.

Note that the m_i orthogonal eigenvectors which were chosen as a basis of the null space $N(P - \lambda_i I)$ are not unique. Any change in these vectors which is done in the same null space spanned by them, merely changes them from one arbitrarily chosen basis to another. Therefore, in the expression for ${}^j\dot{v}_i$ we may drop, without loss of generality, its projection onto its own null space and thus leave in (2.7) only those terms which are orthogonal to the null space of ${}^j v_i$. Consequently,

$${}^j\dot{v}_i = \sum_{\substack{q=1 \\ q\neq i}}^{k} \sum_{p=1}^{m_q} {}^p\alpha_q {}^p v_q. \tag{2.8}$$

The expression $[\dot{P} - {}^j\dot{\lambda}_i I]{}^j v_i$ in (2.5) also belongs to R^n, thus it can be expressed in terms of a basis of R^n, in particular, it can be expressed in terms of the eigenvectors basis as follows:

$$[\dot{P} - {}^j\dot{\lambda}_i I]{}^j v_i = \sum_{p=1}^{m_i} {}^p\beta_i {}^p v_i + \sum_{\substack{q=1 \\ q\neq i}}^{k} \sum_{p=1}^{m_q} {}^p\beta_q {}^p v_q \tag{2.9}$$

where, like in (2.7), the expression was divided into vectors in the null space $N(P - \lambda_i I)$ and vectors in the rest of R^n. Substitution of (2.8) and (2.9) into (2.5) yields

$$[P - \lambda_i I] \sum_{\substack{q=1 \\ q\neq i}}^{k} \sum_{p=1}^{m_q} {}^p\alpha_q {}^p v_q + \sum_{p=1}^{m_i} {}^p\beta_i {}^p v_i + \sum_{\substack{q=1 \\ q\neq i}}^{k} \sum_{p=1}^{m_q} {}^p\beta_q {}^p v_q = 0 \tag{2.10}$$

and since $P {}^p v_q = \lambda_q {}^p v_q$, (2.10) becomes

$$\sum_{\substack{q=1 \\ q\neq i}}^{k} \sum_{p=1}^{m_q} [{}^p\alpha_q(\lambda_q - \lambda_i) + {}^p\beta_q] {}^p v_q + \sum_{p=1}^{m_i} {}^p\beta_i {}^p v_i = 0. \tag{2.11}$$

The left-hand side of (2.11) is a linear combination of linearly independent vectors. Thus, the only way it can equal zero is that each one of the coefficients be equal to zero; that is,

$$ {}^{p}\beta_i = 0 \tag{2.12a} $$

$$ {}^{p}\alpha_q(\lambda_q - \lambda_i) + {}^{p}\beta_q = 0 \qquad \begin{matrix} p = 1, \cdots, m_q \\ q = 1, \cdots, k;\ q \neq i. \end{matrix} \tag{2.12b} $$

Consequently

$$ {}^{p}\alpha_q = \frac{{}^{p}\beta_q}{\lambda_i - \lambda_q} \qquad \begin{matrix} p = 1, \cdots, m_q \\ q = 1, \cdots, k;\ q \neq i. \end{matrix} \tag{2.13} $$

To evaluate ${}^{p}\beta_q$ substitute (2.12a) into (2.9) to obtain

$$ [\dot{P} - {}^{j}\dot{\lambda}_i]{}^{j}v_i = \sum_{\substack{q=1 \\ q\neq i}}^{k} \sum_{p=1}^{m_q} {}^{p}\beta_q {}^{g}v_h^T \cdot {}^{p}v_q. \tag{2.14} $$

and then premultiply (2.14) by ${}^{g}v_h^T$. We choose $h \neq i$, therefore, due to the orthogonality between ${}^{g}v_h$ and ${}^{j}v_i$, we obtain

$$ {}^{g}v_h^T \dot{P}\, {}^{j}v_i = \sum_{\substack{q=1 \\ q\neq i}}^{k} \sum_{p=1}^{m_q} {}^{p}\beta_q {}^{g}v_h^T \cdot {}^{p}v_q. \tag{2.15} $$

The orthonormality of the various eigenvectors also implies

$$ {}^{g}v_h^T \cdot {}^{p}v_q = \begin{cases} 0 \text{ for } p \neq g \text{ or } q \neq h \\ 1 \text{ for } p = g \text{ and } q = h \end{cases} $$

hence, the only term on the right-hand side of (2.15) which does not vanish is the one for which $p = g$ and $q = h$, then

$$ {}^{g}v_h^T \dot{P}\, {}^{j}v_i = {}^{g}\beta_h. $$

A change of indexes yields

$$ {}^{p}\beta_q = {}^{p}v_q^T \dot{P}\, {}^{j}v_i. \tag{2.16} $$

Substitution of ${}^{p}\beta_q$ from (2.16) into (2.13) yields

$$ {}^{p}\alpha_q = \frac{{}^{p}v_q^T \dot{P}\, {}^{j}v_i}{\lambda_i - \lambda_q} \tag{2.17} $$

and a substitution of ${}^{p}\alpha_q$ from (2.17) into (2.8) yields the desired result

$$ {}^{j}\dot{v}_i = \sum_{\substack{q=1 \\ q\neq i}}^{k} \sum_{p=1}^{m_q} \frac{{}^{p}v_q^T \dot{P}\, {}^{j}v_i}{\lambda_i - \lambda_q} {}^{p}v_q. \tag{2.18} $$

Finally, premultiplying (2.14) by ${}^{j}v_i^T$ and using again the orthonormality property of the eigenvectors yields

$$ {}^{j}\dot{\lambda}_i = {}^{j}v_i^T \dot{P}\, {}^{j}v_i. \tag{2.19} $$

Equations (2.18), (2.19) form a set of differential equations for the eigenvalues and eigenvectors of the matrix $P(t)$, given the matrix derivative $\dot{P}(t)$ as a function of these variables.

These results are similar to results known from the theory of perturbations of matrices that depend on a parameter [16], [17]. Derivatives of eigenvalues and eigenvectors were computed also in optimization problems in structural dynamics [18]. In those cases the eigenfactor derivatives were used to compute the change in the eigenfactors of a matrix as a result of a slight perturbation. However, those results were not extended to the case of time derivatives over an unlimited time span and obviously not to the solution of the matrix Riccati equation. Bierman [19] too treated the problem of covariance propagation via its eigenvalues and eigenvectors. His results are similar to the equations presented above, except for the fact that he applied the results to the Lyapunov (linear) equation, while we deal with the Riccati (nonlinear) equation. However, in order to use those results, one must compute $P(t)$ (using its eigenfactors) at each integration step. In addition, the case of very close eigenvalues was not treated. In the following sections we will extend the results given in (2.18) and (2.19) to the solution of the Riccati equation and particular attention will be given to the case of close eigenvalues.

B. Rate of Change of the Eigencomponents of the Riccati Solution

In order to simplify the notation in the ensuing development let us rewrite (2.18) and (2.19) in a different form. To meet this end, let us order the eigencomponents as follows:

$$ {}^{1}\lambda_1, {}^{2}\lambda_1, \cdots, {}^{m_1}\lambda_1, {}^{1}\lambda_2, {}^{2}\lambda_2, \cdots, {}^{m_2}\lambda_2, \cdots, {}^{1}\lambda_k, {}^{2}\lambda_k, \cdots {}^{m_k}\lambda_k $$

$$ {}^{1}v_1, {}^{2}v_1, \cdots, {}^{m_1}v_1, {}^{1}v_2, {}^{2}v_2, \cdots, {}^{m_2}v_2, \cdots, {}^{1}v_k, {}^{2}v_k, \cdots, {}^{m_k}v_k $$

and equate the two sequences, respectively to $\lambda_1, \cdots, \lambda_n$ and $v_1, \cdots, v_n$, then (2.18) can be written as follows:

$$ \dot{v}_i = \sum_{\substack{q=1 \\ \lambda_i \neq \lambda_q}}^{n} \frac{v_q^T \dot{P} v_i}{\lambda_i - \lambda_q} v_q \qquad i = 1, \cdots, n \tag{2.20} $$

and (2.19) can be written as

$$ \dot{\lambda}_i = v_i^T \dot{P} v_i \qquad i = 1, \cdots, n. \tag{2.21} $$

Equations (2.20) and (2.21) which hold for self-adjoint matrices in general, certainly hold for the solution of the following matrix Riccati equation:

$$ \dot{P}(t) = F(t)P(t) + P(t)F(t)^T + Q(t) - P(t)C(t)P(t). \tag{2.22} $$

If we premultiply (2.22) by v_i^T and postmultiply it by v_i, using (2.21), we obtain

$$ \dot{\lambda}_i = v_i^T [\lambda_i F + \lambda_i F^T + Q - \lambda_i^2 C] v_i. \tag{2.23} $$

Now premultiply (2.22) by v_q^T and postmultiply it by v_i to obtain

$$ v_q^T \dot{P} v_i = v_q^T [\lambda_i F + \lambda_q F^T + Q - \lambda_i \lambda_q C] v_i \tag{2.24} $$

which is the numerator expression in the coefficient of v_q in (2.20).

In order to simplify the notation, define the matrix function T_{iq}

$$ T_{iq} \triangleq \lambda_i F + \lambda_q F^T + Q - \lambda_i \lambda_q C. \tag{2.25} $$

Then (2.24) can be written as

$$ v_q^T \dot{P} v_i = v_q^T T_{iq} v_i. \tag{2.26} $$

Next define the scalar function γ_{iq}

$$ \gamma_{iq} \triangleq v_q^T T_{iq} v_i. \tag{2.27} $$

From (2.25), it is easy to see that $T_{iq}^T = T_{qi}$, therefore,

$$ \gamma_{iq} = \gamma_{iq}^T = v_i^T T_{iq}^T v_q = v_i^T T_{qi} v_q, $$

i.e.,

$$ \gamma_{iq} = \gamma_{qi}. \tag{2.28} $$

We define now the matrix Ω, whose element on the qth row and

the ith column is

$$\Omega_{qi}=\begin{cases}\dfrac{\gamma_{iq}}{\lambda_i-\lambda_q} & \text{for } \lambda_i\neq\lambda_q\\ 0 & \text{for } \lambda_i=\lambda_q\end{cases} \tag{2.29}$$

thus Ω is a skew symmetric matrix.

Also define

$$\Lambda=\text{diag}\ (\lambda_1, \lambda_2, \cdots, \lambda_n) \tag{2.30a}$$

$$\Gamma=\text{diag}\ (\gamma_{11}, \gamma_{22}, \cdots, \gamma_{nn}). \tag{2.30b}$$

Using these definitions, (2.20) and (2.21) can now be written as follows:

$$\dot{\Lambda}=\Gamma \tag{2.31a}$$

$$\dot{V}=V\Omega. \tag{2.31b}$$

In order to solve (2.31) for the eigenvalues and eigenvectors of $P(t)$, we have yet to specify the initial conditions, and these are easily found by diagonalizing $P(t)$ at $t = t_o$.

C. Square Root Formulation

Having obtained the set of differential equations (2.31), for the eigenvalues and eigenvectors of P, it is easy to derive an equivalent set in terms of the square root of the eigenvalues. This is done next.

Define the scalar functions $s(t)$ as

$$s_i(t)=\sqrt{\lambda_i(t)} \qquad i=1, 2, \cdots, n. \tag{2.32}$$

Differentiating (2.32) with respect to time yields

$$\dot{s}_i=\frac{1}{2s_i}\dot{\lambda}_i \tag{2.33}$$

and using the elements of (2.31a) in (2.33) we get

$$\dot{s}_i=\frac{\gamma_{ii}}{2s_i}. \tag{2.34}$$

Define

$$S=\text{diag}\ (s_1, s_2, \cdots, s_n) \tag{2.35}$$

then (2.34) can be written as

$$\dot{S}=\frac{1}{2}\Gamma S^{-1}. \tag{2.36}$$

For the nonzero elements of Ω as defined in (2.29) we obtain

$$\Omega_{qi}=\frac{\gamma_{iq}}{s_i^2-s_q^2}=\frac{\gamma_{iq}}{(s_i-s_q)(s_i+s_q)} \tag{2.37}$$

which is then used in (2.31b) to compute V.

The meaning of this formulation will be discussed in Section V.

Examination of (2.31) or their equivalent (2.36), (2.37) reveals that the case of multiple eigenvalues poses no problem at all since the elements of Ω which correspond to those eigenvalues vanish; however, the question arises whether the case of very close eigenvalues can cause numerical difficulties, as it may lead to the computation of very large eigenvector derivatives. This issue is investigated next.

III. The Case of Close Eigenvalues

Examination of (2.29) [or its square root equivalent (2.37)] and (2.31b) reveals that when some of the eigenvalues converge, the computation might run into difficulties, as the magnitudes of the derivatives of the corresponding eigenvectors become intolerably large. A possible solution to the problem is the use of a special program that can handle stiff differential equations. Some known programs that are intended for this use are Gear's DIFSUB [20], and Starner's IMPSUB [21]. The latter program is a modification of the former, and can solve "infinitely stiff" problems in which the derivatives of some of the variables may be missing. However, the use of such special programs may sometimes be inefficient or even impossible. Therefore, we present in this section another solution to the problem of eigenvalue convergence. Our solution consists of a minor modification of the algorithm for computing Ω. The resulting algorithm which removes the difficulty suffers practically no reduction in accuracy. In order to present the remedy to the problem, let us assume that at some stage of the numerical integration of the differential (2.31), one obtains a cluster of r eigenvalues which are close to one another. Consequently, their eigenvector derivatives become intolerably large [see (2.29) or (2.37)]. For the sake of clarity we can assume, with no loss of generality, that all other $n - r$ eigenvalues stay distinct.

Let us order the eigenvalues such that those which belong to the cluster are the first r eigenvalues in the array: $\lambda_1, \lambda_2, \cdots, \lambda_r, \lambda_{r+1}, \cdots, \lambda_n$. Accordingly, we order the corresponding eigenvectors as follows $v_1, v_2, \cdots, v_r, v_{r+1}, \cdots, v_n$. We perform now a direct sum decomposition of the n-dimensional Euclidean space R^n into two subspaces as follows:

$$R^n=S_r\oplus S_{n-r} \tag{3.1}$$

where

S_r is the invariant subspace spanned by the first r eigenvectors (belonging to the clustered eigenvalues),

and

S_{n-r} is the invariant subspace spanned by the rest of the $n-r$ eigenvectors ($S_{n-r}=S_r^{\perp}$, the orthogonal complement of S_r).

For each of the eigenvectors belonging to S_r, we may compute the derivative using (2.31b) as follows:

$$\dot{v}_i=\sum_{q=1}^{n}\Omega_{qi}v_q, \qquad i=1, 2, \cdots, r \tag{3.2}$$

where Ω_{qi} is the q, i element of the matrix Ω which is defined in (2.29). In accordance with the decomposition (3.1), we make a distinction between the eigenvectors belonging to the clustered eigenvalues and all other eigenvectors. Accordingly, (3.2) is written as

$$\dot{v}_i=\dot{v}_i^{(r)}+\dot{v}_i^{(n-r)}, \qquad i=1, 2, \cdots, r \tag{3.3}$$

where

$$\dot{v}_i^{(r)}=\sum_{q=1}^{r}\Omega_{qi}v_q \tag{3.4a}$$

and

$$\dot{v}_i^{(n-r)}=\sum_{q=r+1}^{n}\Omega_{qi}v_q. \tag{3.4b}$$

Obviously, $\dot{v}_i^{(r)}\in S_r$ and $\dot{v}_i^{(n-r)}\in S_{n-r}$.

We now proceed by making the following proposition.

Proposition 3.1: The rate of change of the subspace S_r is independent of the components of those derivatives, of its spanning eigenvectors, which belong to S_r. That is, S_r is independent of the eigenvector derivatives $\dot{v}_1^{(r)}, \dot{v}_2^{(r)}, \cdots, \dot{v}_r^{(r)}$.

To prove this proposition we note that, in general, the rate of change of any subspace of R^n may be expressed as a function of

its basis vectors, and their derivatives (i.e., a description of their change). In our case this claim means that the rate of change of the subspace S_r can be expressed as a function of the following vectors:

$$\{v_1, v_2, \cdots, v_r\} \in S_r,$$

$$\{\dot{v}_1^{(r)}, \dot{v}_2^{(r)}, \cdots, \dot{v}_r^{(r)}\} \in S_r$$

and

$$\{\dot{v}_1^{(n-r)}, \dot{v}_2^{(n-r)}, \cdots, \dot{v}_r^{(n-r)}\} \in S_{n-r}.$$

Now, in passing from (2.7) to (2.8) it was argued that a change in the eigenvectors, which is performed in the null space spanned by them, merely changes the basis of the null space and not the null space itself. Similarly, since $\{\dot{v}_i^{(r)}\}_{i=1}^r \in S_r$, this set of vectors expresses the change of the eigenvector set $\{v_i\}_{i=1}^r$ in S_r, and thus it does not contribute to the change of S_r itself. That is, the set $\{v_i^{(r)}\}_{i=1}^r$ represents the rotation of the set $\{v_i\}_{i=1}^r$ in the subspace spanned by it.

As a consequence of this argument, we may express the rate of change of S_r as a function of $\{v_i\}_{i=1}^r$ and $\{\dot{v}_i^{(n-r)}\}_{i=1}^r$ alone, and hence Proposition 3.1 is proved.

Next we define the orthogonal projector E_i as the matrix representation of the linear transformation of any vector $x \in R^n$ into its orthogonal projection on the null space $N(P - \lambda_i I)$; that is

$$E_i x \in N(P - \lambda_i I) \qquad \forall x \in R^n. \tag{3.5}$$

For any subspace $L \subset R^n$ for which an orthonormal basis is available, the projector E_L from R^n onto L may be computed by

$$E_L = \sum_{j=1}^{\sigma} x_j x_j^T \tag{3.6}$$

where $x_1, x_2, \cdots, x_\sigma$ are a set of basis vectors of L [22]. In particular, we may use (3.6) to compute the orthogonal projector upon the subspace S_r:

$$\bar{E}_r = \sum_{i=1}^{r} v_i v_i^T. \tag{3.7}$$

The projector $\bar{E}_r$ can also be expressed in terms of the individual projectors on the subspaces spanned by each of the eigenvectors $\{v_i\}_{i=1}^r \in S_r$ [23]

$$\bar{E}_r = \sum_{i=1}^{r} E_i \tag{3.8}$$

where, obviously, $E_i = v_i v_i^T$.

The following proposition relates the rates of change of the subspace S_r and $\bar{E}_r$, the orthogonal projector upon it.

Proposition 3.2: The rate of change of the projector $\bar{E}_r$ is independent of those components of the derivatives of the eigenvectors spanning S_r, which belong to S_r, i.e., it is independent of $\dot{v}_1^{(r)}, \dot{v}_2^{(r)}, \cdots, \dot{v}_r^{(r)}$.

To prove this proposition, we note that $\bar{E}_r$ carries any vector $x \in R^n$ into its orthogonal projection $\bar{E}_r x$ in S_r. From Proposition 3.1 we know that the subspace S_r does not change as a result of the action of the components of the derivatives $\{\dot{v}_i^{(r)}\}_{i=1}^r$. The orthogonal projection is unique, hence, it is also not affected by $\{\dot{v}_i^{(r)}\}_{i=1}^r$. We therefore conclude that the rate of change of $\bar{E}_r$ is independent of $\{\dot{v}_i^{(r)}\}_{i=1}^r$.

Next we use spectral decomposition in order to decompose $P(t)$ as follows:

$$P(t) = \sum_{i=1}^{n} \lambda_i E_i. \tag{3.9}$$

Denote the mean value of the clustered eigenvalues by $\bar{\lambda}_r$, then (3.9) can be written as

$$P(t) = \sum_{i=1}^{r} (\lambda_i - \bar{\lambda}_r) E_i + \bar{\lambda}_r \bar{E}_r + \sum_{i=r+1}^{n} \lambda_i E_i \tag{3.10}$$

where use of (3.8) was made. Note that in (3.10), λ_i, $\bar{\lambda}_r$, E_i, and $\bar{E}_r$ are all time varying. Also note that the clustered eigenvalues are assumed to be close to one another but none of them are identical. As the clustered eigenvalues are assumed to be very close to one another we can make use of the following proposition.

Proposition 3.3: Define the approximation index d as follows

$$d \triangleq \max_{1 \le i,j \le r} |\lambda_i - \lambda_j| \tag{3.11}$$

and approximate $P(t)$ by $P_c(t)$ where

$$P_c(t) \triangleq \bar{\lambda}_r \bar{E}_r + \sum_{i=r+1}^{n} \lambda_i E_i \tag{3.12}$$

then

$$\|P - P_c\| \le rd \tag{3.13}$$

where the matrix norm used here is the spectral norm.

This proposition can be easily proven since

$$\|P - P_c\| \le \sum_{i=1}^{r} |\lambda_i - \bar{\lambda}_r| \le rd. \tag{3.14}$$

From this proposition it is clear that

$$\|P - P_c\| \xrightarrow[d \to 0]{} 0. \tag{3.15}$$

As a consequence, we conclude that when the clustered eigenvalues are very close to one another, we may neglect the term $\Sigma_{i=1}^r (\lambda_i - \bar{\lambda}_r) E_i$ on the right-hand side of (3.10), at a price of a negligible error in the computation of $P(t)$. That is, we may use the projector $\bar{E}_r$ instead of the individual projectors E_i in order to compute $P(t)$. Now, from Proposition 3.2 we know that the components $\{\dot{v}_i^{(r)}\}_{i=1}^r$ have no effect on the change of $\bar{E}_r$ in time, and from Proposition 3.3 we see that $P(t)$ can be approximated by a decomposition which contains $\bar{E}_r$ but not the individual E_i of v_i which are in S_r. Consequently, the approximation of $P(t)$ which is based on $\bar{E}_r$ (3.12) is independent of the components $\{\dot{v}_i^{(r)}\}_{i=1}^r$. We therefore conclude that when the eigenvalues are clustered, these components may be approximated in some way, such that there is no need to use the expression (2.29) which is the source of the difficulty. This approximation will have very little effect on the accuracy of the computation. For the implementation of the last result, we use a modified version of (2.29)

$$\Omega_{qi} = \begin{cases} \dfrac{\gamma_{iq}}{\lambda_i - \lambda_q} & \text{for } \left|\dfrac{\gamma_{iq}}{\lambda_i - \lambda_q}\right| < \Omega_{\max} \\ 0 & \text{for } \lambda_i = \lambda_q \\ \Omega_{\max} \operatorname{sign}\left\{\dfrac{\gamma_{iq}}{\lambda_i - \lambda_q}\right\} & \text{for } \left|\dfrac{\gamma_{iq}}{\lambda_i - \lambda_q}\right| \ge \Omega_{\max} \end{cases} \tag{3.16}$$

where $\Omega_{\max}$ has to be determined prior to the computation according to the ability of the software used and the accuracy sought.

In the following section where we present numerical results, we also present a simple numerical example of a case of close eigenvalues which demonstrate the results of this section.

IV. Numerical Examples

In this section we present results of two examples. The purpose of this presentation is 1) to demonstrate that the algorithm is

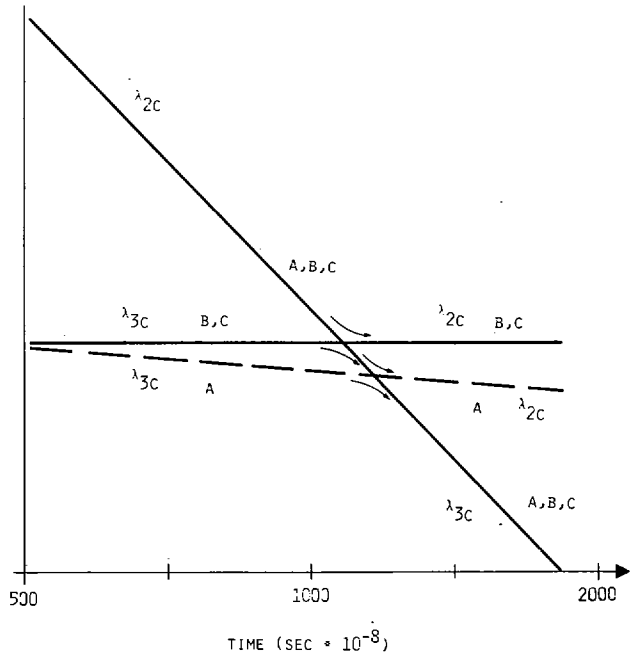


Fig. 1. The behavior of λ_{2c} and λ_{3c} in the vicinity of t_c for the three solutions. *A*: The direct solution in single precision. *B*: The direct solution in double precision. *C*: The new algorithm in single precision.

working, 2) to show the higher accuracy of the algorithm vis-a-vis the direct solution, and 3) to use an example by way of demonstrating the behavior of the eigenvectors when eigenvalues approach one another.

A. Example 1

In this example of order three we mainly wish to demonstrate the behavior of the eigenvectors which belong to two eigenvalues as the latter approach one another. As a byproduct, the accuracy of the algorithm is demonstrated also.

The coefficient matrices of the matrix Riccati equation were chosen as follows [see (1.1)]:

$$F = \begin{bmatrix} 0.5E-3 & 0.2 & 0.2E-1 \\ 0.1 & 0.2E-3 & 0. \\ 0.1E-1 & 0. & 0.1E-3 \end{bmatrix} \quad Q = \text{diag } [1.\ 2.\ 3.] \quad C = \begin{bmatrix} 10. & 10. & 10. \\ 10. & 10. & 10. \\ 10. & 10. & 10. \end{bmatrix}$$

and the initial condition was

$$P_o = \begin{bmatrix} 10.00858 & 0.4760068E-2 & 0.47860067E-2 \\ 0.4760068E-2 & 7.500974 & -2.496704 \\ 0.47860067E-2 & -2.496704 & 7.501056 \end{bmatrix}.$$

The eigenvalues of the latter matrix were $\lambda_1 = 5.00428963$, $\lambda_2 = 9.9977026$, and $\lambda_3 = 10.0085878$. At time $t_c = 0.1037E-4$ the eigenvalues of the solution matrix $P(t_c)$ were $\lambda_{1c} = 4.9984$, λ_{2c}, $\lambda_{3c} = 9.9977$; that is, two of the eigenvalues became identical. (It should be noted that since the solution was a numerical one and was performed on a discrete computing machine, the equality was only to within a finite, albeit small, difference.) Note that the computation at t_c is trivial [see (2.29)]. The time point t_c is listed here as a reference point only. We are interested in the computation in the vicinity of t_c and not at t_c itself. The matrix Riccati equation was solved on an IBM 3081D machine using the Runge–Kutta–Verner fifth- and sixth-order integration method which is implemented in a member of the IMSL library (Edition 9.1). The initial step size of the integration was set to be $\Delta t = 10^{-8}$ s. (The integration routine automatically divides the initial step size to meet a given error measure. The error measure was chosen to be equal in all runs of this example.)

Three different runs were performed. In the first run the direct solution of (1.1) was computed in single precision and then the eigenfactors of the solution matrix were computed at the end of each time step. The behavior of λ_{2c} and λ_{3c} about t_c is shown in Fig. 1. In the second run the same solution method was used, only here it was computed in double precision. The behavior of λ_{2c} and λ_{3c} in this case is also shown in Fig. 1. Finally, the algorithm presented in this paper was used in the third run which was performed in single precision. The results of this run are also shown in Fig. 1. The value of Ω_{max} which was chosen in the third run [see (3.17)] was $\Omega_{max} = 10^7$. We note that the accuracy achieved using the direct solution in double precision resembles the accuracy achieved when using the new algorithm in single precision (this observation is based on the computation of λ_{3c} before t_c and of λ_{2c} after t_c). This quality exhibited by the new algorithm is a distinct characteristic of square root algorithms.

To analyze the behavior near t_c of the two eigenvectors which belong to the eigenvalues λ_{2c} and λ_{3c} we turn to (2.29) and (2.31b). Define

$$\omega_1 = \frac{\gamma_{23}}{\lambda_2 - \lambda_3} \quad \omega_2 = \frac{\gamma_{31}}{\lambda_3 - \lambda_1} \quad \omega_3 = \frac{\gamma_{12}}{\lambda_1 - \lambda_2} \tag{4.1}$$

then from (2.28), (2.29), and (2.31b)

$$[\dot{v}_1\ \dot{v}_2\ \dot{v}_3] = [v_1\ v_2\ v_3] \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{4.2}$$

from which we obtain

$$\dot{v}_1 = \omega_3 v_2 - \omega_2 v_3 \tag{4.3a}$$

$$\dot{v}_2 = -\omega_3 v_1 + \omega_1 v_3 \quad (4.3b)$$

$$\dot{v}_3 = \omega_2 v_1 - \omega_1 v_2. \quad (4.3c)$$

From (4.1) it is realized that at the vicinity of t_c, where the second and third eigenvalues approach one another, ω_{1c} (ω_1 near t_c) takes a very large value. Since all the eigenvectors are of unit length, it is concluded from (4.3) that

$$\dot{v}_{2c} \sim \omega_{1c} v_{3c} \quad (4.4a)$$

$$\dot{v}_{3c} \sim \omega_{1c} v_{2c} \quad (4.4b)$$

v_{1c}, v_{2c}, and v_{3c} denote, correspondingly, v_1, v_2, and v_3 at the vicinity of t_c and due to the value of ω_{1c}, the magnitude of the latter two vectors is enormous. At the same time the magnitude of $\dot{v}_1$ is bounded. From the analysis presented in Section III and from (4.4), it is concluded that at the vicinity of t_c, v_{2c} and v_{3c} rotate perpendicularly to one another at a very fast rate in the subspace S_r which they span. This subspace is obviously a plane which practically does not change its orientation in the three-dimensional space of the problem. The reason why the plane practically does not change its orientation stems from the fact that although v_{2c} and v_{3c} rotate rapidly in the close vicinity of t_c, the change in v_{1c}, the normal to the plane, is negligible throughout the short time interval in which the fast change in v_{2c} and v_{3c} occurs. This is so since the value of the rate of change of v_{1c} is bounded. The behavior of ω_{1c} in this example is shown in Fig. 2.

Fig. 3 describes the behavior of the rotation of v_{2c} and v_{3c} as a function of time in the plane S_r. The angle α is defined in plane S_r from some reference line. We note that besides a short interval about t_c, the motion of v_{2c} is quite regular. As explained earlier, although v_{2c} and v_{3c} change erratically at a small time interval near t_c, their exact value is irrelevant as long as they span the plane which is perpendicular to v_{1c} and the latter is indeed accomplished when the new algorithm is used as outlined in the preceding sections.

Finally, in order to assure that the preceding results are independent of the integration method, the example was repeated using Gear's stiff method integration routine of the same library. This repetition yielded similar results.

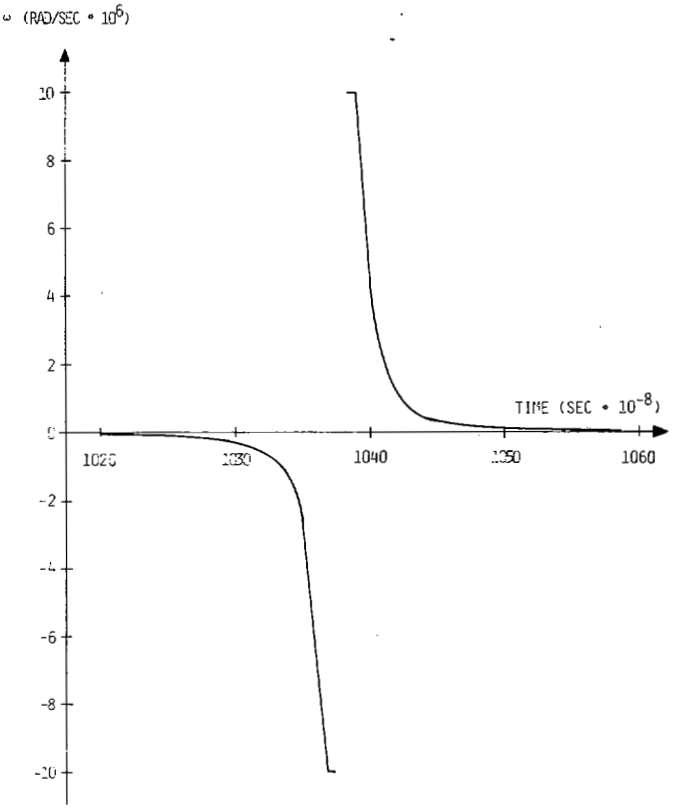


Fig. 2. The behavior of ω_{1c} as a function of time near t_c.

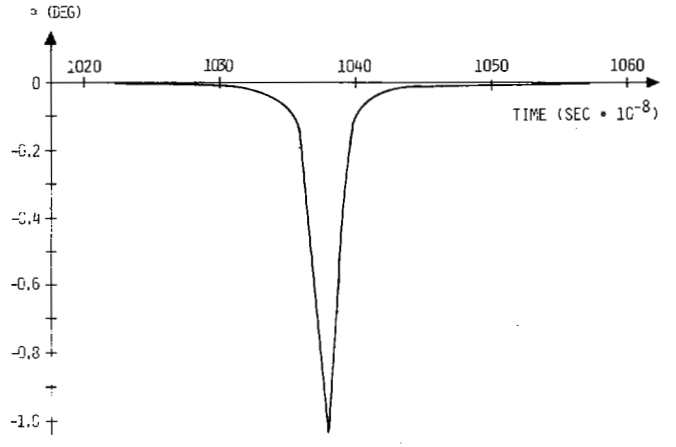


Fig. 3. The angular change of v_{2c} and v_{3c} in the plane S_r at times close to t_c.

B. Example 2

In this example we wish to demonstrate the accuracy of the new algorithm. Here we considered a continuous filtering problem where

$$P_o = \text{diag } [0.02,\ 0.02,\ 0.02,\ 0.214E-10,\ 0.214E-10]$$

$$F = \begin{bmatrix} 0 & 10. & 0 & 0 & 0 \\ -0.1587E-6 & 0. & 0.6E-4 & 1. & 1. \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$Q = \text{diag } [0,\ 0,\ 0,\ 0,\ 0.713E-15]$$

$$H = [1,\ 0,\ 0,\ 0,\ 0]$$

$$R^{-1} = 0.15E+6.$$

For the filtering problem $C = H^T R^{-1} H$. The initial step size was $\Delta t = 0.1$ s and the integration routine used was Gear's stiff method integration routine of the library mentioned in Example 1. Note that in the time invariant case, equation (1.1) has the closed form solution [3]

$$P(t) = AB^{-1} \quad (4.5a)$$

$$A = Z_{21} + Z_{22} P_o \qquad B = Z_{11} + Z_{12} P_o \quad (4.5b)$$

$$Z = \begin{bmatrix} Z_{11} & Z_{12} \\ Z_{21} & Z_{22} \end{bmatrix} \quad (4.5c)$$

$$Z = \exp \begin{bmatrix} -F^T & C \\ Q & F \end{bmatrix} (t - t_o). \quad (4.5d)$$

When the direct solution of (1.1) was computed in single precision some of the eigenvalues of the solution matrix became negative and the solution blew up at time $t = 221.4$ s. Moreover, even the closed-form solution (4.5) blew up since the matrix B was nearly singular. However, when the new algorithm was used to solve (1.1), a solution was obtained at any desired point prior or beyond 221.4 s. At time $t_f = 100$ s, for example, the result was

$$P_{t_f} = \begin{bmatrix} 0.33E-05 & 0.77E-07 & 0.82E-04 & 0.15E-08 & 0.11E-08 \\ 0.77E-07 & 0.30E-08 & 0.49E-05 & 0.87E-10 & 0.65E-10 \\ 0.82E-04 & 0.49E-05 & 0.17E-01 & -0.45E-07 & -0.35E-07 \\ 0.15E-08 & 0.87E-10 & -0.45E-07 & 0.21E-10 & -0.63E-12 \\ 0.11E-08 & 0.65E-10 & -0.35E-07 & -0.63E-12 & 0.15E-10 \end{bmatrix}. \quad (4.6)$$

To demonstrate that this is indeed the solution, the value of P_{tf} was computed at $t_f - \Delta t$ and $t_f + \Delta t$ and then a computation of $\dot{P}_{tf}$ was performed using

$$\dot{P}_{tf} = \frac{P_{tf+\Delta t} - P_{tf-\Delta t}}{2\Delta t}. \tag{4.7}$$

Next the value of P_{tf} was used on the right-hand side of (1.1) to compute $\dot{P}^*_{tf}$ and then an accuracy index was computed as follows:

$$e = \|\dot{P}_{tf} - \dot{P}^*_{tf}\| \tag{4.8}$$

where the norm was the Euclidean norm. This computation yielded the result

$$e = 0.4324E - 06. \tag{4.9}$$

The new algorithm yields the correct solution when the direct solution and the closed-form solution fail. This is again a distinct characteristic of square root algorithms.

Finally, it should be noted that the divergence of the direct solution is characteristic in certain problematic filtering problems. Such a case was purposely chosen for this example in order to demonstrate the performance of the new algorithm.

V. Concluding Remarks

In this paper we have presented an algorithm which yields, directly, the eigenfactors of the solution of the matrix Riccati equation (1.1). The algorithm consists of a set of differential equations which express the first time derivative of the eigenfactors in terms of the eigenfactors themselves and do not use the solution matrix of (1.1). The algorithm handles distinct as well as multiple eigenvalues. The algorithm is basically a square root algorithm. We presented a version which is analogous to Bierman's UDU^T algorithm where V, the orthogonal matrix of eigenvectors, is analogous to U, and Λ, the diagonal matrix of eigenvalues, is analogous to D. We presented also a full square root version in which S rather than Λ is used where S is a diagonal matrix of the positive square root of the eigenvalues.

We presented two examples in which we used the full square root version. Repeating these examples with the UDU^T-like version, yielded similar conclusions. The examples showed that the accuracy achieved using the new algorithm in single precision resembles the accuracy obtained when using the direct solution in double precision.

We developed a more concise version which used the orthogonality of the eigenvectors; however, that version had a lower accuracy and was, therefore, abandoned.

Finally, we used the eigenfactor solution to successfully develop a discrete square root filter algorithm which will be presented in a subsequent paper.

Acknowledgment

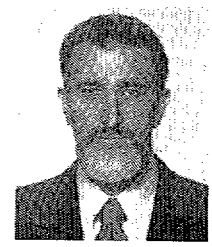
The authors wish to thank Dr. L. Rodman of the School of Mathematics, Tel-Aviv University, who is one of the authors of [14], for his helpful discussion.

References

[1] M. Athans and R. L. Falb, *Optimal Control: An Introduction to the Theory and Its Application*. New York: McGraw-Hill, 1966.
[2] A. E. Bryson, Jr. and Y. C. Ho, *Applied Optimal Control*. Washington, DC: Hemisphere, 1974.
[3] H. Kwakernaak and R. Sivan, *Linear Optimal Control Systems*. New York: Wiley, 1972.
[4] A. H. Jazwinski, *Stochastic Processes and Filtering Theory*. New York: Academic, 1970.
[5] A. Gelb, Ed., *Applied Optimal Estimation*. Cambridge, MA: M.I.T. Press, 1974.
[6] S. F. Schmidt, "Computation techniques in Kalman filtering," *Theory and Application of Kalman Filtering*, Advisory Group for Aerospace Research and Development, AGARDograph 139 (AD 704 306), Feb. 1970.
[7] P. G. Kaminski, A. E. Bryson, Jr., and J. F. Schmidt, "Discrete square root filtering: A survey of current techniques," *IEEE Trans. Automat. Contr.*, vol. AC-16, pp. 727–736, Dec. 1971.
[8] P. S. Maybeck, *Stochastic Models, Estimation, and Control*, Vol. 1, New York: Academic, 1979.
[9] G. J. Bierman, *Factorization Methods for Discrete Sequential Estimation*. New York: Academic, 1977.
[10] A. Andrews, "A square root formulation of the Kalman covariance equations," *AIAA J.*, vol. 6, no. 6, pp. 1165–1166, June 1968.
[11] B. D. Tapley and C. Y. Choe, "An algorithm for propagating the square root covariance matrix in triangular form," *IEEE Trans. Automat. Contr.*, vol. AC-21, pp. 122–123, Feb. 1976.
[12] B. D. Tapley and J. G. Peters, "Sequential estimation algorithm using a continuous UDU^T covariance factorization," *J. Guidance Contr.*, vol. 3, pp. 326–331, July–Aug. 1980.
[13] M. Morf, B. Levy, and T. Kailath, "Square root algorithms for the continuous time linear least-squares estimation problem," *IEEE Trans. Automat. Contr.*, vol. AC-23, pp. 907–911, Oct. 1978.
[14] I. Gohberg, P. Lancaster, and L. Rodman, *Matrix Polynomials*. New York: Academic, 1982.
[15] L. A. Zadeh and C. A. Desoer, *Linear System Theory: The State Space Approach*. New York: McGraw-Hill, 1963.
[16] R. Bellman, *Introduction to Matrix Analysis*. New York: McGraw-Hill, 1970.
[17] J. M. Franklin, *Matrix Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1968.
[18] R. L. Fox and M. P. Kapoor, "Rates of change of eigenvalues and eigenvectors," *AIAA J.*, vol. 6, pp. 2426–2429, Dec. 1968.
[19] G. J. Bierman, "Covariance propagation via its eigenvalues and eigenvectors," *AIAA J.*, vol. 8, no. 5, pp. 958–959, May 1980.
[20] C. W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations*. Englewood Cliffs, NJ: Prentice-Hall, 1971.
[21] J. Starner, "Numerical solution of implicit differential-algebraic equations," Ph.D. dissertation, Univ. New Mexico, Albuquerque, NM, 1976.
[22] A. Ben-Israel and T. N. E. Greville, *Generalized Inverses: Theory and Applications*. New York: Wiley, 1974.
[23] P. R. Halmos, *Finite Dimensional Vector Spaces*. Princeton, NJ: Van-Nostrand, 1958.

Yaakov Oshman was born in Israel on November 4, 1953. He received the B.Sc. degree (summa cum laude) in aeronautical engineering from the Technion—Israel Institute of Technology, Haifa, Israel, in 1975.

From 1975 to 1981 he was with the Israeli Air Force, where he worked in the areas of structural dynamics and flutter analysis and flight testing. In 1981 he started his graduate work in the Department of Aeronautical Engineering, Technion—Israel Institute of Technology, where he is currently pursuing a D.Sc. program. His research interests include numerical solutions of the matrix Riccati differential equation, square root, and reduced order filtering.



Itzhack Y. Bar-Itzhack (M'73–SM'84) was born in Israel on August 19, 1937. He received the B.Sc. and M.Sc. degrees in electrical engineering from the Technion—Israel Institute of Technology, Haifa, Israel, in 1961 and 1964, respectively, and the Ph.D. degree in electrical engineering from the University of Pennsylvania, Philadelphia, in 1968.

From 1961 to 1964 he was a Teaching and Research Assistant in the field of automatic control at the Technion—Israel Institute of Technology. In 1965 he became a Research Assistant at the Moore School of Electrical Engineering, University of Pennsylvania, where he did work on strapdown inertial navigation systems. From 1968 to 1971 he was a member of the Technical Staff at Bellcomm, Inc., Washington, DC, where he worked on the Saturn V longitudinal stability problem and on the Lunar Roving Vehicle navigation system. In 1971 he joined the Department of Aeronautical Engineering, Technion—Israel Institute of Technology where he is Professor and Head of the Technion Space Research Institute. During the 1977–1978 academic year he spent his sabbatical with The Analytic Sciences Corporation (TASC), Reading, MA, where he worked in research and development of multisensor integrated navigation. His research interests include inertial navigation, estimation, and guidance.

Dr. Bar-Itzhack is a member of Sigma Xi.