Pergamon

Computers Math. Applic. Vol. 33, No. 4, pp. 69–79, 1997
Copyright©1997 Elsevier Science Ltd
Printed in Great Britain. All rights reserved
0898-1221/97 $17.00 + 0.00

PII: S0898-1221(97)00008-4

Inversion Formulas and Linear Complexity Algorithm for Diagonal Plus Semiseparable Matrices

Y. EIDELMAN AND I. GOHBERG
School of Mathematical Sciences
Raymond and Beverly Sackler Faculty of Exact Sciences
Tel-Aviv University, Ramat-Aviv 69978, Israel

(Received July 1996; accepted August 1996)

Abstract—Matrices represented as a sum of diagonal and semiseparable ones are considered here. These matrices belong to the class of structured matrices which arises in numerous applications. Fast $O(N)$ algorithms for their inversion were developed before under additional restrictions which were a source of instability. Our aim is to eliminate these restrictions and to develop reliable and stable numerical algorithms. In this paper, the case of semiseparable matrices of order one is considered.

Keywords—Linear complexity algorithm, Semiseparable matrices.

1. INTRODUCTION

In the present paper, we consider an $N \times N$ matrix R of the form

$$R = D + S, \tag{1.1}$$

where $D = \operatorname{diag}\{d_k, 1 \le k \le N\}$ is a diagonal matrix and $S = \{s_{i,j}\}_{i,j=1}^N$ is a semiseparable matrix of order one.

By definition, matrix $A = \{a_{ij}\}_{i,j=1}^N$ is called semiseparable of order n if for some vectors $g_k = \{g_k(i)\}_{i=1}^N$, $h_k = \{h_k(i)\}_{i=1}^N$, $p_k = \{p_k(i)\}_{i=1}^N$, $q_k = \{q_k(i)\}_{i=1}^N$, $k = 1, \dots, n$,

$$a_{ij} = \begin{cases} \sum_{k=1}^n g_k(i)h_k(j), & 1 \le i < j \le N, \\ \sum_{k=1}^n p_k(i)q_k(j), & 1 \le j < i \le N, \\ 0, & i = j. \end{cases}$$

Thus, the elements of S in (1.1) are of the form

$$s_{ij} = \begin{cases} g(i)h(j), & 1 \le i < j \le N, \\ 0, & i = j, \\ p(i)q(j), & 1 \le j < i \le N, \end{cases} \tag{1.2}$$

where $p = \{p(i)\}_{i=1}^N$, $q = \{q(i)\}_{i=1}^N$, $g = \{g(i)\}_{i=1}^N$, $h = \{h(i)\}_{i=1}^N$ are given N-dimensional vectors. In other words, the matrix S is composed of the upper triangular part of the matrix $gh^\top$ of rank at most one, and from the lower triangular part of another matrix $pq^\top$ also of rank at most one.

Typeset by $\mathcal{A}_\mathcal{M}\mathcal{S}$-TEX

Our aim is to obtain for the matrix of the form (1.1),(1.2), a numerically reliable inversion $O(N)$ algorithm which can also be used for solving linear systems $Rx = y$ at cost $O(N)$ arithmetic operations.

Probably the first time the linear complexity algorithm for inversion of such matrices was suggested in [1,2] was the assumption that the matrix R is strongly regular, i.e., all its leading minors are nonvanishing. In a paper of Gohberg and Kaashoek [3], the matrices of the form (1.1),(1.2) arose as input-output maps for discrete linear systems. In [3], an inversion formula was obtained by the assumption that the values $\delta_k = d_k - g(k)h(k)$, which are called external coefficients, are nonvanishing. The formula obtained in [3] also admits a $O(N)$ algorithm of numerical realization. In the present paper, we limit ourselves to the case $n = 1$, for which we obtain a rather simple linear complexity algorithm which is reliable and stable without any additional restrictions on the matrix R except its invertibility. The case of a semiseparable matrix of any order n will be considered in a later publication.

We propose here a modification of Gohberg-Kaashoek formula valid for arbitrary values of δ_k including zero. This modification is used to obtain a reliable inversion algorithm. Even in the case of a sum of a diagonal matrix and a matrix of rank 1, the results obtained seem to be new. A wide set of computer experiments was performed. In these experiments, special attention was paid to the cases of close to zero leading minors and close to zero external coefficients δ_k.

2. REPRESENTATION OF THE MATRIX

Here we consider an invertible $N \times N$ matrix R of the form

$$R = D + S, \tag{2.1}$$

where $D = \operatorname{diag}\{d_k, 1 \le k \le N\}$ is a diagonal matrix and $S = \{s_{ij}\}_{i,j=1}^N$ is a semiseparable matrix of order one, i.e., the elements of S in (2.1) are of the form

$$s_{ij} = \begin{cases} g(i)h(j), & 1 \le i < j \le N, \\ 0, & i = j, \\ p(i)q(j), & 1 \le j < i \le N, \end{cases} \tag{2.2}$$

where $p = \{p(i)\}_{i=1}^N$, $q = \{q(i)\}_{i=1}^N$, $g = \{g(i)\}_{i=1}^N$, $h = \{h(i)\}_{i=1}^N$ are given N-dimensional vectors.

Matrix R may be represented as

$$R = (F + L) + gh^{\top}, \tag{2.3}$$

where

$$F = \operatorname{diag}\{\delta_i\}_{i=1}^N, \qquad \delta_i = d_i - g(i)h(i),$$
$$L(i,j) = \begin{cases} 0, & i \le j, \\ p(i)q(j) - g(i)h(j), & 1 \le j < i \le N. \end{cases}$$

Put

$$B_k = \begin{bmatrix} q(k) \\ h(k) \end{bmatrix}, \qquad C_k = [p(k) \ \ -g(k)]. \tag{2.4}$$

Then one can write the elements of L as

$$L(i,j) = \begin{cases} 0, & i \le j, \\ C_iB_j, & 1 \le j < i \le N. \end{cases}$$

Matrix R is represented as a sum of lower triangular matrix and matrix of rank 1. Moreover, matrix L in (2.3) is semiseparable of order 2.

Similar to (2.3), one can represent

$$R = (E + T) + pq^{\mathsf{T}}, \tag{2.5}$$

where

$$E = \operatorname{diag}\{l_i\}_{i=1}^N, \qquad l_i = d_i - p(i)q(i),$$
$$T(i,j) = \begin{cases} -p(i)q(j) + g(i)h(j) = -C_iB_j, & 1 \le i < j \le N, \\ 0, & i \ge j. \end{cases}$$

Suppose that in (2.3) and (2.5), $\delta_i \neq 0$, $l_i \neq 0$, for $i = 1, \dots N$. In this case, we have the relations

$$R^{-1} = (F + L)^{-1} - (F + L)^{-1} g \left(U^{\times}\right)^{-1} h^{\mathsf{T}} (F + L)^{-1}, \tag{2.6}$$

where

$$U^{\times} = 1 + h^{\mathsf{T}}(F + L)^{-1} g,$$

and

$$R^{-1} = (E + T)^{-1} - (E + T)^{-1} p \left(V^{\times}\right)^{-1} q^{\mathsf{T}} (E + T)^{-1}, \tag{2.7}$$

where

$$V^{\times} = 1 + q^{\mathsf{T}}(E + T)^{-1} p,$$

and the equality

$$\det R = U^{\times} \left(\prod_{i=1}^N \delta_i\right) = V^{\times} \left(\prod_{i=1}^N l_i\right). \tag{2.8}$$

Following the Gohberg-Kaashoek method, we reduce (2.6),(2.7) to a more convenient form for further computations. Let us introduce the family of 2×2 matrices $U_{kj}, k \ge j$ and $V_{kj}, k \le j$ by the formulas

$$A_k = I - \frac{1}{\delta_k} B_k C_k, \qquad U_{kj} = A_{k-1} \dots A_j, \ k > j, \qquad U_{kk} = I, \tag{2.9}$$
$$G_k = I + \frac{1}{l_k} B_k C_k, \qquad V_{kj} = G_k \dots G_{j-1}, \ k < j, \qquad V_{jj} = I. \tag{2.10}$$

It is not difficult to prove that the matrix $(F + L)^{-1}$ is given by the relations

$$(F + L)^{-1} = \operatorname{diag}\left\{\frac{1}{\delta_i}\right\}_{i=1}^N + \left[L_{ij}^{\times}\right]_{i,j=1}^N,$$

where

$$L_{ij}^{\times} = \begin{cases} -\frac{1}{\delta_i} C_i U_{i,j+1} B_j \frac{1}{\delta_j}, & 1 \le j < i \le N, \\ 0, & i \le j. \end{cases} \tag{2.11}$$

Further, from (2.11) it follows

$$-(F + L)^{-1} g = (F + L)^{-1} C K_0 = \operatorname{col}\left(\frac{1}{\delta_j} C_j U_{j,1}\right)_{j=1}^N, \tag{2.12}$$

where $K_0 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$, and

$$h^{\mathsf{T}}(F + L)^{-1} = K_0^{\mathsf{T}} B (F + L)^{-1} = K_0^{\mathsf{T}} \operatorname{row}\left(U_{N+1,j+1} B_j \frac{1}{\delta_j}\right)_{j=1}^N. \tag{2.13}$$

By virtue of (2.6), (2.12), (2.13), we obtain

$$U^{\times} = K_0^{\top} U_{N+1,1} K_0. \tag{2.14}$$

Thus, we obtain the formula

$$R^{-1} = (F+L)^{-1} + R^{\times}, \tag{2.15}$$

where

$$R^{\times}_{ij} = \frac{1}{\delta_i}\left(C_i U_{i,1} K_0\right)\left(U^{\times}\right)^{-1}\left(K_0^{\top} U_{N+1,j+1} B_j\right)\frac{1}{\delta_j}, \qquad 1 \le i,j \le N.$$

A similar formula is derived from (2.7) and (2.10). One can represent the matrix $(E+T)^{-1}$ as

$$(E+T)^{-1} = \operatorname{diag}\left\{\frac{1}{l_i}\right\}_{i=1}^{N} + \left[T^{\times}_{ij}\right]_{i,j=1}^{N},$$

$$T^{\times}_{ij} = \begin{cases} \dfrac{1}{l_i} C_i V_{i+1,j} B_j \dfrac{1}{l_j}, & 1 \le i < j \le N, \\ 0, & 1 \le j \le i \le N. \end{cases}$$

Further set $L_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, and then

$$\begin{gathered} (E+T)^{-1} p = (E+T)^{-1} C L_0 = \operatorname{col}\left(\frac{1}{l_i} C_i V_{i+1,N+1}\right)_{i=1}^{N}, \\ q^{\top}(E+T)^{-1} = L_0^{\top} \operatorname{row}\left(V_{1,j} B_j \frac{1}{l_j}\right)_{j=1}^{N}, \\ V^{\times} = L_0^{\top} V_{1,N+1} L_0. \end{gathered} \tag{2.16}$$

Thus, we have

$$R^{-1} = (E+T)^{-1} + Q^{\times}, \tag{2.17}$$

where

$$Q^{\times}_{i,j} = -\frac{1}{l_i}\left(C_i V_{i+1,N+1} L_0\right)\left(V^{\times}\right)^{-1}\left(L_0^{\top} V_{1,j} B_j\right)\frac{1}{l_j}.$$

Based on (2.15),(2.17), we are able to formulate the following assertion.

THEOREM 1. *Let R be an invertible matrix of form (2.1),(2.2). Let $\delta_k = d_k - g(k)h(k) \neq 0$, $l_k = d_k - p(k)q(k) \neq 0$.*

Then the following inversion formula holds:

$$R^{-1}(i,j) = \begin{cases} \dfrac{1}{\delta_i}\left(C_i U_{i,1} K_0\right)\left(U^{\times}\right)^{-1}\left(K_0^{\top} U_{N+1,j+1} B_j\right)\dfrac{1}{\delta_j}, & 1 \le i < j \le N, \\ \dfrac{1}{\delta_i} + \dfrac{1}{\delta_i}\left(C_i U_{i,1} K_0\right)\left(U^{\times}\right)^{-1}\left(K_0^{\top} U_{N+1,i+1} B_i\right)\dfrac{1}{\delta_i}, & i = j, \\ -\dfrac{1}{l_i}\left(C_i V_{i+1,N+1} L_0\right)\left(V^{\times}\right)^{-1}\left(L_0^{\top} V_{1,j} B_j\right)\dfrac{1}{l_j}, & 1 \le j < i \le N, \end{cases} \tag{2.18}$$

where the following notations are used:

$$B_k = \begin{bmatrix} q(k) \\ h(k) \end{bmatrix}, \qquad C_k = [p(k) \;\; -g(k)], \qquad K_0 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \qquad L_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix},$$

$$U_{kj} = A_{k-1} \dots A_j, \quad k > j, \qquad A_k = I - \frac{1}{\delta_k} B_k C_k, \qquad U_{kk} = I,$$

$$V_{kj} = G_k \dots G_{j-1}, \quad k < j, \qquad G_k = I + \frac{1}{l_k} B_k C_k, \qquad V_{jj} = I,$$

$$U^{\times} = K_0^{\top} U_{N+1,1} K_0, \qquad V^{\times} = L_0^{\top} V_{1,N+1} L_0,$$

and

$$\det R = U^\times \left(\prod_{i=1}^{N} \delta_i\right) = V^\times \left(\prod_{i=1}^{N} l_i\right).$$

The representation (2.18) is a modification of the Gohberg-Kaashoek formula which was obtained in [3] in a more general situation.

3. INVERSION FORMULA

The next aim is to derive from Theorem 1, the inversion formula for matrix R in the case where there are no restrictions $\delta_k \neq 0$ and $l_k \neq 0$.

At first, we consider the case $i < j$.

Set

$$\begin{aligned} A_k^\times &= \delta_k A_k = \delta_k I - B_k C_k, && 1 \le k \le N, \\ Z_k &= A_{k-1}^\times \dots A_1^\times K_0, \quad k \ge 2, && Z_1 = K_0, && (3.1) \\ S_k &= K_0^\top A_N^\times \dots A_k^\times, \quad k \le N, && S_{N+1} = K_0^\top, && (3.2) \\ \Delta_{mk} &= \delta_m \dots \delta_k, \quad m \le k, && \Delta_{m,m-1} = 1. && (3.3) \end{aligned}$$

Then

$$U_{i,1} K_0 = Z_i \frac{1}{\Delta_{1,i-1}}, \quad i \ge 1, \qquad K_0^\top U_{N+1,j+1} = \frac{1}{\Delta_{j+1,N}} S_{j+1}, \quad j \le N,$$

and moreover, by virtue of (2.14)

$$U^\times = S_k Z_k \frac{1}{\Delta_{1,N}}, \qquad 1 \le k \le N+1,$$

and by virtue of (2.8)

$$\det R = S_k Z_k \neq 0, \qquad 1 \le k \le N+1.$$

Thus, we obtain

$$R^{-1}(i,j) = C_i Z_i \left(S_1 Z_1\right)^{-1} \Delta_{i+1,j-1} S_{j+1} B_j, \qquad i < j. \tag{3.4}$$

Similarly for $i > j$, we set

$$\begin{aligned} G_k^\times &= l_k G_k = l_k I + B_k C_k, && 1 \le k \le N, \\ X_k &= G_k^\times \dots G_N^\times L_0, \quad k \le N, && X_{N+1} = L_0, && (3.5) \\ Y_k &= L_0^\top G_1^\times \dots G_{k-1}^\times, \quad k \ge 2, && Y_1 = L_0^\top, && (3.6) \\ \gamma_{ik} &= l_i \dots l_k, \quad i \ge k, && \gamma_{i-1,1} = 1. && (3.7) \end{aligned}$$

Then,

$$V_{i+1,N+1} L_0 = X_{i+1} \frac{1}{\gamma_{N,i+1}}, \quad i \le N, \qquad L_0^\top V_{1,j} = \frac{1}{\gamma_{j-1,1}} Y_j, \quad j \ge 1,$$

and by virtue of (2.16)

$$V^\times = \frac{1}{\gamma_{N,1}} Y_k X_k, \qquad 1 \le k \le N+1,$$

and by virtue of (2.8)

$$\det R = Y_k X_k \neq 0, \qquad 1 \le k \le N+1.$$

Thus,

$$R^{-1} = -C_i X_{i+1} \left(Y_1 X_1\right)^{-1} \gamma_{i-1,j+1} Y_j B_j, \qquad i > j. \tag{3.8}$$

For the diagonal, we have

$$\begin{aligned} R^{-1}(i,i) &= \frac{1}{\delta_i} + C_i Z_i \frac{1}{\Delta_{1,i}} \left(\frac{1}{\Delta_{i+1,N}} S_{i+1} \left(I - \frac{1}{\delta_i} B_i C_i \right) Z_i \frac{1}{\Delta_{1,i-1}} \right)^{-1} \frac{1}{\Delta_{i,N}} S_{i+1} B_i \\ &= \frac{1}{\delta_i} + \frac{1}{\delta_i} \frac{(C_i Z_i)(S_{i+1} B_i)}{\delta_i S_{i+1} Z_i - (S_{i+1} B_i)(C_i Z_i)}, \end{aligned}$$

and thus,

$$R^{-1}(i,i) = \frac{S_{i+1} Z_i}{\delta_i S_{i+1} Z_i - (S_{i+1} B_i)(C_i Z_i)}. \tag{3.9}$$

Note that the denominator in the latter fraction equals $\det R$.

The formulas (3.4), (3.8), (3.9) lead to the following theorem.

THEOREM 2. *Let R be an invertible of matrix of the form (2.1),(2.2).*
Introduce the vectors and the numbers

$$B_k = \begin{bmatrix} q(k) \\ h(k) \end{bmatrix}, \qquad C_k = [p(k) \; -g(k)], \qquad K_0 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \qquad L_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix},$$
$$\delta_k = d_k - g(k)h(k), \qquad l_k = d_k - p(k)q(k), \qquad 1 \le k \le N,$$
$$\Delta_{mk} = \delta_m \dots \delta_k, \quad m \le k, \qquad \Delta_{m,m-1} = 1, \qquad \gamma_{ik} = l_i \dots l_k, \quad i \ge k, \qquad \gamma_{i-1,i} = 1,$$

and by using them, define recursively

$$\begin{array}{lll} Z_1 = K_0, & Z_{k+1} = (\delta_k I - B_k C_k) Z_k, & k = 1 \dots N, \\ S_{N+1} = K_0^{\mathrm{T}}, & S_k = S_{k+1} (\delta_k I - B_k C_k), & k = N \dots 1, \\ Y_1 = L_0^{\mathrm{T}}, & Y_{k+1} = Y_k (l_k I + B_k C_k), & k = 1 \dots N, \\ X_{N+1} = L_0, & X_k = (l_k I + B_k C_k) X_{k+1}, & k = N \dots 1. \end{array} \tag{3.10}$$

Then, $U_0 = S_i Z_i = Y_i X_i = \det R \neq 0$, $1 \le i \le N+1$, and the inverse matrix R^{-1} is given by the relations

$$R^{-1}(i,j) = \begin{cases} v(i)(U_0)^{-1} \Delta_{i+1,j-1} u(j), & i < j, \\ \alpha_i (U_0)^{-1}, & i = j, \\ -r(i)(U_0)^{-1} \gamma_{i-1,j+1} s(j), & i > j, \end{cases} \tag{3.11}$$

where

$$v(k) = C_k Z_k, \qquad u(k) = S_{k+1} B_k, \qquad s(k) = Y_k B_k, \qquad r(k) = C_k X_{k+1}, \qquad \alpha_k = S_{k+1} Z_k.$$

4. THE INVERSION ALGORITHM

The numerical experiments showed that direct computations by formula (3.11) for large N leads to overflow. It occurs because of large values of the elements Z_k, S_k, X_k, Y_k, which are the products of a large number of the factors $\delta_k I - B_k C_k$ and $l_k I + B_k C_k$. This effect may be overcome by an appropriate scaling. One can introduce the scaling coefficients for two-dimensional vector W by the rule

$$\eta(W) = \max(|W(1)|, |W(2)|), \qquad \beta(W) = \begin{cases} \dfrac{1}{\eta(W)}, & \eta(W) > 1, \\ 1, & \eta(W) \le 1, \end{cases}$$

and set

$$\begin{array}{lll} Z'_1 = K_0, & Z'_{k+1} = \beta(Z'_k)(\delta_k I - B_k C_k) Z'_k, & k = 1 \dots N, \\ S'_{N+1} = K_0^{\mathsf{T}}, & S'_k = \beta(S'_{k+1}) S'_{k+1}(\delta_k - B_k C_k), & k = N \dots 1, \\ Y'_1 = L_0^{\mathsf{T}}, & Y'_{k+1} = \beta(Y'_k) Y'_k (l_k I + B_k C_k), & k = 1 \dots N, \\ X'_{N+1} = L_0, & X'_k = \beta(X'_{k+1})(l_k I + B_k C_k) X'_{k+1}, & k = N \dots 1. \end{array}$$

Then, it is easy to see that the elements of the recursion (3.10) are transformed as follows:

$$\begin{array}{ll} Z_k = Z'_k \left(\beta(Z'_{k-1}) \dots \beta(Z'_1)\right)^{-1}, & S_k = S'_k \left(\beta(S'_{k+1}) \dots \beta(S'_{N+1})\right)^{-1}, \\ Y_k = Y'_k \left(\beta(Y'_{k-1}) \dots \beta(Y'_1)\right)^{-1}, & X_k = X'_k \left(\beta(X'_{k+1}) \dots \beta(X'_{N+1})\right)^{-1}. \end{array}$$

Moreover, to transform the elements of the inversion formula (3.11) in the case $i < j$, one can use the representations

$$v(i) = C_i Z'_i \left(\beta(Z'_{i-1}) \dots \beta(Z'_1)\right)^{-1}, \qquad u(j) = \left(\beta(S'_{j+2}) \dots \beta(S'_{N+1})\right)^{-1} S'_{j+1} B_j,$$
$$U_0 = \left(\beta(S'_{i+2}) \dots \beta(S'_{N+1})\right)^{-1} S'_{i+1} Z'_{i+1} \left(\beta(Z'_i) \dots \beta(Z'_1)\right)^{-1},$$

and obtain

$$R^{-1}(i,j) = C_i Z'_i \beta(Z'_i) \left(S'_{i+1} Z'_{i+1}\right)^{-1} \Delta_{i+1,j-1} \beta(S'_{i+2}) \dots \beta(S'_{j+1}) S'_{j+1} B_j, \qquad i < j. \tag{4.1}$$

Similarly, for $i > j$ using the representations

$$s(j) = \left(\beta(Y'_{j-1}) \dots \beta(Y'_1)\right)^{-1} Y'_j B_j, \qquad r(i) = C_i X'_{i+1} \left(\beta(X'_{i+2}) \dots \beta(X'_{N+1})\right)^{-1},$$
$$U_0 = \left(\beta(Y'_{i-1}) \dots \beta(Y'_1)\right)^{-1} Y'_i X'_i \left(\beta(X'_{i+1}) \dots \beta(X'_{N+1})\right)^{-1},$$

one can obtain

$$R^{-1}(i,j) = -C_i X'_{i+1} \beta(X'_{i+1}) \left(Y'_i X'_i\right)^{-1} \gamma_{i-1,j+1} \beta(Y'_{i-1}) \dots \beta(Y'_j) Y'_j B_j, \qquad i > j, \tag{4.2}$$

and for $i = j$ using

$$\alpha_i = \left(\beta(S'_{i+2}) \dots \beta(S'_{N+1})\right)^{-1} S'_{i+1} Z'_i \left(\beta(Z'_{i-1}) \dots \beta(Z'_1)\right)^{-1},$$
$$U_0 = \left(\beta(S'_{i+1}) \dots \beta(S'_{N+1})\right)^{-1} S'_i Z'_i \left(\beta(Z'_{i-1}) \dots \beta(Z'_1)\right)^{-1},$$

we obtain

$$R^{-1}(i,i) = \frac{\beta(S'_{i+1})\left(S'_{i+1} Z'_i\right)}{(S'_i Z'_i)}. \tag{4.3}$$

Thus (4.1)–(4.3) lead to another variant of an inversion formula:

$$R^{-1}(i,j) = \begin{cases} v'(i) \Delta'_{i+1,j-1} u'(j), & i < j, \\ \alpha'_i, & i = j, \\ -r'(i) \gamma'_{i-1,j+1} s'(j), & i > j, \end{cases} \tag{4.4}$$

where the following notations are used:

$$v'(i) = C_i Z'_i \beta(Z'_i) \left(S'_{i+1} Z'_{i+1}\right)^{-1}, \qquad u'(j) = S'_{j+1} B_j \beta(S'_{j+1}),$$
$$r'(i) = C_i X'_{i+1} \beta(X'_{i+1}) \left(Y'_i X'_i\right)^{-1}, \qquad s'(j) = Y'_j B_j \beta(Y'_j),$$
$$\alpha'_i = \frac{\beta(S'_{i+1})\left(S'_{i+1} Z'_i\right)}{(S'_i Z'_i)},$$
$$\Delta'_{mk} = \delta_m \beta(S'_{m+1}) \dots \delta_k \beta(S'_{k+1}), \quad m \le k, \qquad \Delta'_{m,m-1} = 1,$$
$$\gamma'_{ik} = l_i \beta(Y'_i) \dots l_k \beta(Y'_k), \quad i \ge k, \qquad \gamma'_{i-1,i} = 1.$$

To compute the elements of relation (4.4), the following algorithm is used.

1. Start with $Z'_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$, $Y'_1 = (1\ 0)$ and compute recursively for $i = 1, \dots, N$,

$$B_i = \begin{bmatrix} q(i) \\ h(i) \end{bmatrix}, \qquad C_i = [p(i)\ \ -g(i)], \qquad \delta_i = d_i - g(i)h(i), \qquad l_i = d_i - p(i)q(i),$$
$$\tilde{v}(i) = C_i Z'_i \beta(Z'_i), \qquad s'(i) = Y'_i B_i \beta(Y'_i),$$
$$Z'_{i+1} = \delta_i Z'_i \beta(Z'_i) - B_i \tilde{v}(i), \qquad Y'_{i+1} = Y'_i l_i \beta(Y'_i) + s'(i) C_i.$$

2. Start with $S'_{N+1} = (0\ 1)$, $X'_{N+1} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and compute recursively for $i = N, \dots, 1$,

$$u'(i) = S'_{i+1} B_i \beta(S'_{i+1}), \qquad \tilde{r}(i) = C_i X'_{i+1} \beta(X'_{i+1}),$$
$$S'_i = S'_{i+1}\delta_i \beta(S'_{i+1}) - u'(i) C_i, \qquad X'_i = \beta(X'_{i+1}) l_i X'_{i+1} + B_i \tilde{r}(i),$$
$$v'(i) = \tilde{v}(i)(S_{i+1}Z_{i+1})^{-1}, \qquad r'(i) = \tilde{r}(i)(Y_i X_i)^{-1}.$$

3. Compute recursively for $i = 1, \dots, N$,

$$\alpha'_i = \frac{\beta(S'_{i+1})(S'_{i+1}Z'_i)}{(S'_i Z'_i)}.$$

This algorithm requires $18N$ additions/subtractions and $48N$ multiplications/divisions.

5. SOLUTION OF LINEAR SYSTEMS

The solution of the system of linear algebraic equations $Rx = y$, where R is the matrix of the form (2.1),(2.2) and y is a given vector from $\mathbb{C}^N$ is found as $x = x_L + x_D + x_U$, $x_L = R_L y$, $x_D = R_D y$, $x_U = R_U y$ and R_L, R_D, R_U are correspondingly lower triangular, diagonal, and upper triangular parts of the matrix R^{-1} which is given by formula (4.4).

For x_L, we have $x_L(1) = 0$ and for $i \geq 2$,

$$x_L(i) = -r'(i) z_i,$$

where

$$z_i = \sum_{j=1}^{i-1} \gamma'_{i-1,j+1} s'(j) y(j),$$

moreover, z_i satisfies the recursive relations

$$z_{i+1} = \sum_{j=1}^{i} \gamma'_{i,j+1} s'(j) y(j) = \beta(Y'_i) l_i z_i + s'(i) y(i).$$

Similar relations hold for the upper triangular part, i.e., for the x_U.

Thus, the following algorithm is obtained.

1. Start with $x_L(1) = 0$, $z_1 = 0$ and for $i = 2, \dots, N$, compute recursively,

$$z_i = \beta(Y'_{i-1}) l_{i-1} z_{i-1} + s'(i-1) y(i-1),$$
$$x_L(i) = -r'(i) z_i.$$

2. Compute recursively for $i = 1, \dots, N$,

$$x_D(i) = \alpha'_i y(i).$$

3. Start with $x_U(N) = 0$, $w_N = 0$ and for $i = N-1, \dots, 1$, compute recursively,

$$w_i = \beta\left(S'_{i+2}\right)\delta_{i+1}w_{i+1} + u'(i+1)y(i+1),$$
$$x_U(i) = v'(i)w_i.$$

4. Compute the solution x

$$x = x_L + x_D + x_U.$$

This algorithm requires $7N$ additions/subtractions and $15N$ multiplications. Thus, the total amount of operations to solve the system is $22N$ additions/subtractions and $58N$ multiplications/divisions.

6. NUMERICAL EXPERIMENTS

We performed a large number of computer experiments with the algorithm designed in Sections 4 and 5 to investigate its behavior in floating point arithmetic and to compare it with other available algorithms. We solved linear systems $Rx = y$ for random values of input data p, q, g, h, D, y using the following algorithms.

(1) GKK Gohberg-Kailath-Koltracht algorithm from [1].
(2) GK Realization of Gohberg-Kaashoek formula (2.15).
(3) GE Algorithm presented in Sections 4 and 5.

All of the algorithms (1)–(3) were implemented in the system MATLAB, version 4.2 with unit round-off error 2.2204×10^{-16}. The accuracy of the obtained solutions was estimated by the relations

$$\varepsilon = \frac{||x - x_G||}{||x_G||}, \qquad \varepsilon_y = \frac{||Rx - y||}{||y||},$$

where x is the solution obtained by the corresponding algorithm, x_G is the solution obtained by the Gauss method using the standard MATLAB function which we assume to be exact. The values of the input data were obtained by using the random-function. In each case, the conditional number $k_2(R)$ of the original matrix was also computed.

In all performed experiments, the values of p, q, g, h, y were chosen at random in the range of 0 to 10. To obtain close to zero leading minors or coefficients δ_k, we controlled the values of the diagonal part D.

1. In the first series of experiments, the values of D were taken from the range of 0 to 100 which corresponds to the same scale of all elements of the matrix R. The results are presented in the Table 1.

Table 1.

		GKK		GK		GE	
N	$K_2(R)$	ε	ε_y	ε	ε_y	ε	ε_y
5	50.53	4.89e−16	1.35e−15	4.11e−16	4.81e−15	2.43e−15	2.01e−15
40	1.66e+04	2.92e−14	9.22e−14	5.46e−11	6.85e−10	4.90e−14	1.88e−12
100	2.44e+05	7.50e−14	1.41e−11	3.81e−07	8.47e−04	2.00e−14	2.61e−11
160	9.37e+03	6.05e−15	1.95e−14	3.78e+12	6.37e+13	7.27e−15	1.84e−13
240	5.39e+05	1.84e−13	1.49e−13	2.10e+13	6.76e+18	1.86e−13	1.50e−11

Such a rough error in the GK algorithm for large N is accounted for by the fact that the solution obtained by formula (2.15) is a sum $x = (D+S)^{-1}y + R^{\times}y$, where elements of the matrices $(D+S)^{-1}$, $R^{\times}$ for large N are the products of a large number of the factors $I - (1/\delta_k)B_kC_k$. As a result, we obtain, and it is confirmed directly, a sum of two very large items of different signs that leads to a large computing error.

Table 2.

N	$K_2(R)$	GKK		GK		GE	
		ε	ε_y	ε	ε_y	ε	ε_y
5	9.36	4.60e−17	1.56e−16	6.46e−16	3.35e−16	1.67e−16	2.34e−16
40	7.83e+02	2.94e−15	2.42e−15	3.27e−15	3.25e−15	6.20e−15	6.37e−15
100	6.57e+02	2.02e−14	3.46e−14	7.28e−15	2.42e−14	3.04e−15	6.83e−15
160	1.68e+04	1.80e−14	7.17e−13	7.47e−13	2.29e−11	8.23e−15	9.52e−14
200	2.05e+03	3.05e−15	1.79e−15	4.01e−13	2.60e−12	2.62e−15	5.45e−15

2. In the second series of experiments, we chose the diagonal elements in the range of 0 to 1000, which led to the greater values of $\delta_k = d_k - g(k)h(k)$, decrease of the elements $(1/\delta_k)B_kC_k$, and accordingly to improvement of the GK algorithm performance. The results are presented in Table 2.

3. The following series of experiments was performed to investigate the stability of considering algorithms for δ_k close to zero. We obtained the values δ_2, δ_4 close to zero by the suitable choice of the diagonal elements d_2, d_4. The other elements d_k were chosen similar to the previous series in the range of 0 to 1000. The results are presented in Table 3. They show the stability of GKK and GE algorithms; just as in the GK algorithm, the considerable reduction of accuracy is observed.

Table 3.

N	δ_2	δ_4	$K_2(R)$	GKK		GK		GE	
				ε	ε_y	ε	ε_y	ε	ε_y
5	1e−3	1	169.23	2.86e−16	1.71e−16	1.36e−09	3.54e−10	1.06e−15	6.52e−16
80	1e−3	1	2.34e+03	1.63e−15	1.07e−13	4.52e−07	1.02e−05	2.74e−15	8.69e−15
200	1e−3	1	3.56e+03	4.74e−15	1.88e−14	4.27e−07	7.08e−06	4.53e−15	3.94e−14
5	1e−5	1	160.09	1.31e−16	4.55e−16	1.35e−08	7.13e−08	3.00e−16	3.60e−15
60	1e−5	1	2.66e+04	7.16e−15	3.37e−14	4.70e−10	1.51e−07	2.35e−15	3.59e−13
200	1e−5	1	6.01e+03	7.90e−15	4.03e−15	0.0011	0.0132	1.05e−14	3.92e−14
5	1e−3	1e−3	144.87	3.01e−16	1.11e−15	9.47e−08	4.44e−07	9.98e−16	3.25e−15
80	1e−3	1e−3	1.07e+03	4.09e−15	4.18e−15	0.0033	0.0418	1.83e−15	1.16e−15
160	1e−3	1e−3	9.94e+03	5.03e−16	1.04e−14	5.95e−07	2.52e−05	5.35e−15	3.77e−15
5	1e−5	1e−5	164.79	4.48e−16	2.06e−16	0.0076	0.0048	2.99e−15	7.77e−16
160	1e−5	1e−5	1.25e+04	1.45e−14	9.01e−14	4.92	724.71	6.44e−15	1.74e−14

Table 4.

N	γ_2	γ_4	$K_2(R)$	GKK		GK		GE	
				ε	ε_y	ε	ε_y	ε	ε_y
5	1e−3	1	116.16	2.59e−12	4.23e−12	1.57e−16	2.62e−16	2.94e−16	4.04e−16
80	1e−3	1	1.83e+03	3.66e−13	4.71e−12	1.46e−12	2.70e−11	2.74e−15	1.35e−14
160	1e−3	1	1.67e+03	3.39e−11	4.86e−11	4.14e−14	1.28e−13	1.02e−15	4.23e−15
5	1e−5	1	271.49	5.86e−12	1.58e−10	1.83e−15	3.98e−15	1.82e−15	1.49e−14
5	1	1e−5	542.31	5.82e−08	3.43e−07	3.35e−16	9.92e−16	1.34e−15	1.54e−14
80	1	1e−5	2.21e+03	6.76e−09	5.78e−09	1.44e−13	2.50e−12	8.86e−16	1.57e−15
160	1	1e−5	3.11e+03	1.96e−08	4.36e−07	2.55e−13	9.49e−12	2.60e−15	1.82e−14
5	1e−3	1e−3	271.75	7.02e−11	4.06e−10	4.56e−16	2.71e−15	4.66e−15	3.97e−14
80	1e−3	1e−3	2.79e+03	3.45e−11	2.46e−11	3.40e−13	6.07e−12	4.00e−15	2.08e−15
160	1e−3	1e−3	2.75e+03	3.54e−12	7.03e−11	1.42e−13	4.20e−12	3.85e−15	3.47e−14
5	1e−5	1e−5	271.61	9.31e−09	5.40e−08	1.49e−15	2.42e−15	9.10e−16	1.16e−14
80	1e−5	1e−5	2.79e+03	1.51e−08	8.78e−09	4.12e−13	4.35e−12	8.23e−15	8.82e−15
200	1e−5	1e−5	1.91e+03	1.27e−10	1.41e−09	1.81e−09	2.70e−08	2.44e−15	4.80e−15

4. In the final series, we investigated the behavior of algorithms for leading minors close to zero. Let us denote a leading minor of order k of the matrix R by $|R_k|$ and let $\gamma_k = (|R_k|/|R_{k-1}|)$. Only the elements γ_k are used in the GKK algorithm (see [1]). We obtained the elements γ_2 and γ_4 close to zero by the suitable choice of the elements d_2, d_4. The other elements d_k was given similar to cases 2 and 3. The results are presented in Table 4.

The accuracy of the GKK algorithm worsened with the approach of γ_2, γ_4 to zero, the GK algorithm turns out unstable for the large N, and the GE algorithm demonstrates a stable behavior.

Thus, in all experiments performed, the GE algorithm demonstrates a stable behavior.

REFERENCES

1. I. Gohberg, T. Kailath and I. Koltracht, Linear complexity algorithms for semiseparable matrices, *Integral Equations and Operator Theory* **8**, 780–804 (1985).
2. I. Gohberg, T. Kailath and I. Koltracht, A note on diagonal innovation matrices, *Acoustics, Speech and Signal Processing* **7**, 1068–1069 (1987).
3. I. Gohberg and M.A. Kaashoek, Time varying linear systems with boundary conditions and integral operators, 1. The transfer operator and its properties, *Integral Equations and Operator Theory* **7**, 325–391 (1984).