Numer Algor (2008) 47:15–34
DOI 10.1007/s11075-007-9144-9

ORIGINAL PAPER

A fast method to block-diagonalize a Hankel matrix

Skander Belhaj

Received: 15 May 2007 / Accepted: 6 November 2007 /
Published online: 12 December 2007

Abstract In this paper, we consider an approximate block diagonalization algorithm of an $n \times n$ real Hankel matrix in which the successive transformation matrices are upper triangular Toeplitz matrices, and propose a new fast approach to compute the factorization in $O(n^2)$ operations. This method consists on using the revised Bini method (Lin et al., Theor Comp Sci 315: 511–523, 2004). To motivate our approach, we also propose an approximate factorization variant of the customary fast method based on Schur complementation adapted to the $n \times n$ real Hankel matrix. All algorithms have been implemented in Matlab and numerical results are included to illustrate the effectiveness of our approach.

Keywords Hankel matrix · Block diagonalization · Triangular Toeplitz matrix · Schur complementation

S. Belhaj
Laboratoire de Mathématiques, CNRS UMR 6623, UFR des Sciences et Techniques, Université de Franche-Comté, 25030 Besançon cedex, France

S. Belhaj (✉)
Laboratoire LAMSIN, Ecole Nationale d'Ingénieurs de Tunis, Université Tunis El-Manar, BP 37, 1002 Tunis Belvédère, Tunisia
e-mail: skander.belhaj@univ-fcomte.fr

1 Introduction

Let

$$H = \begin{pmatrix} h_1 & h_2 & \cdots & h_n \\ h_2 & h_3 & \cdots & h_{n+1} \\ \vdots & \vdots & \ddots & \vdots \\ h_n & h_{n+1} & \cdots & h_{2n-1} \end{pmatrix} \tag{1}$$

be an $n \times n$ nonsingular, real Hankel matrix. The special structure of Hankel matrices arises in a number of applications in scientific computing, coding, signal and image processing, control, algebraic computing and partial differential equations... [15].

In this paper we consider the problem of a block factorization of an $n \times n$ real Hankel matrix in which the successive transformation matrices are upper triangular Toeplitz matrices. Such a diagonalization returns a block diagonal matrix

$$D = A^t H A = \begin{pmatrix} {}^lH_1 & & & \\ & {}^lH_2 & & \\ & & \ddots & \\ & & & {}^lH_n \end{pmatrix} \tag{2}$$

where every block ${}^lH_j$, for $j = 1, 2, ..., n$ is a lower Hankel matrix and A is an upper triangular matrix.

This problem is studied by Phillips [18], Rissanen [19], Kung [12], Gragg and Lindquist [6], Pal and Kailath [17], Bini and Pan [5], Bultheel and Van Barel [4] and recently by Ben Atti and Diaz-Toca [2].

Among early investigators are Phillips [18] and Rissanen [19]. Later Kung [12] and Gragg and Lindquist [6] considered (2) in the context of partial realization problems. Many relations between Hankel matrices and partial realizations are found in [6, 10, 11]. These techniques are closely related to the theory of moments [1], which has applications in many areas, including computational geometry [7]. Besides, Pal and Kailath [17] considered the factorization of Hankel matrices (2) as a particular application of computing rank profile and inertia [14]. Recently, Ben Atti and Diaz-Toca [2] introduced (2), using "truncated" polynomial division which requires $4n^2 - 7n + 5$ arithmetic operations.

Although the algorithm described in [2] seems to be the fastest, it presents some instability. In fact, when we perturb the Hankel matrix, the block diagonalization gives 1×1 blocks instead of $m \times m$ blocks. Therefore, we introduce the new concept of the approximate block diagonalization to resolve this problem.

Such a diagonalization returns an approximate block diagonal matrix

$$D_\varepsilon = A_\epsilon^t H A_\epsilon = \begin{pmatrix} lH_{\epsilon_1} & \Theta_{1,2} & \cdots & \Theta_{1,n} \\ \Theta_{2,1} & lH_{\epsilon_2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \Theta_{n-1,n} \\ \Theta_{n,1} & \cdots & \Theta_{n,n-1} & lH_{\epsilon_n} \end{pmatrix} \tag{3}$$

where every block lH_{ϵ_j}, for $j = 1, 2, ..., n$ is an approximate lower Hankel matrix, where all entries of $\Theta_{j,k}$, for $j, k = 1, 2, ..., n$ and $j \neq k$ are very close to zero and A_ϵ is an approximate upper triangular matrix.

Our contribution is the proposal of a fast $O(n^2)$ method for the approximate block diagonalization of $n \times n$ real Hankel matrix for an approximate way based on using the revised Bini's method [13] for computing the inverse of an upper triangular Toeplitz matrix. We also compare our approach to an approximate factorization variant of the customary fast method based on Schur complementation (see References [5, 8, 15] for instance) adapted to an $n \times n$ real Hankel matrix.

The paper is organized as follows: In Section 2, we give some notations and lemmas. Our approximate block diagonalization of an $n \times n$ real Hankel is given in Section 3. In Section 4, we propose an approximate factorization variant of the customary fast method based on Schur complementation. We also introduce a comparison with our approach. In Section 5, some numerical examples are given to put in evidence the potential advantages of our approximate decomposition with respect to the classical approximate factorization method, in terms of numerical stability and in terms of computational cost. Finally, a summary and future work are presented in Section 6 to complete the paper.

2 Notation and lemma

We will use the following notations:

- $H(S) \in \mathbb{R}^{n\times n}$denotes the Hankel matrix associated to a list S of length $(2n-1)$. This means that the first row is given by the first n terms of S and the last column is given by the last n terms of S.
- $lH(S) \in \mathbb{R}^{n\times n}$denotes the Hankel triangular matrix (with respect to the antidiagonal) associated to a list S of length $(2n-1)$ such that the last column is defined by S.
- $H(S; m; n) \in \mathbb{R}^{m\times n}$denotes the Hankel matrix associated to a list S of length $(m+n-1)$. The first row is given by the first n terms of S and the last column is given by the last m terms of S.
- $T(S; m; n) \in \mathbb{R}^{m\times n}$denotes the Toeplitz matrix associated to a list S of length $(m+n-1)$. The first row is given by the first m terms of S and the last column is given by the last n terms of S.

- $uT(S) \in \mathbb{R}^{n\times n}$ denotes the upper Toeplitz triangular matrix associated to a list S such that the first row is defined by S.
- Let $p \in \mathbb{N}$. Let $\Sigma_p \in \mathbb{R}^{p\times p}$, $\Sigma_p = \left[\varepsilon_{jk}\right]_{j,k=1}^{p}$, where all entries of Σ_p are zero except that $\varepsilon_{j+k,j} = \varepsilon_k$ for $j, k = 1, 2, ..., n-1$.
- $J_p = lH(1, \underbrace{0, ..., 0}_{p-1}),\ p \in \mathbb{N}$.
- Given $P \in \mathbb{R}^{n\times m}$, $\widetilde{P} = J_m P^t J_n$.
- $T(S; m; n) = J_m H(S; m; n)$.
- $uT(S) = J_l lH(S)$, where l is the length of L.
- Let $a \in \mathbb{R}$. Let $\mu > 0$, $V(a, \mu) = (a-\mu; a+\mu)$ is a neighborhood of a.

Lemma 2.1 *Let $n \in \mathbb{N}^*$. Let $h = H(h_1, ..., h_{2n-1})$ be a square Hankel matrix of order n. Suppose that $h_j = \varepsilon_j$ with $\varepsilon_j \in V(0, \mu)$ for $j = 1, 2, ..., p-1$ and $h_p \notin V(0, \mu)$. Then h has the form*

$$h = H\left(\varepsilon_1, ..., \varepsilon_{p-1}, h_p, ..., h_{2n-1}\right) = \begin{pmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{pmatrix}, \tag{4}$$

where

$$h_{11} = lH\left(h_p, ..., h_{2p-1}\right) + J_p\Sigma_p, \quad h_{22} = H\left(h_{2p+1}, ..., h_{2n-1}\right),$$

$$h_{12} = H\left(h_{p+1}, ..., h_{n+p-1}; p; n-p\right), \quad h_{21} = h_{12}^t.$$

We can successively construct from h the following two matrices:

- A square lower Hankel triangular matrix H of order $(2n-p)$,

$$\mathcal{H} = lH\left(h_p, ..., h_{2n-1}\right) = \begin{pmatrix} 0 & 0 & H_{13} \\ 0 & h_{11} & h_{12} \\ H_{31} & h_{12}^t & h_{22} \end{pmatrix}, \tag{5}$$

  where $H_{31} = H_{13} = lH(h_p, ..., h_{n-1})$
- A square upper triangular Toeplitz matrix T,

$$T = J_{2n-r}\mathcal{H} = uT\left(h_p, ..., h_{2n-1}\right) = \begin{pmatrix} t_{11} & t_{12} & t_{13} \\ 0 & t_{22} & t_{23} \\ 0 & 0 & t_{33} \end{pmatrix}, \tag{6}$$

  where $t_{kj} = JH_{3-i+1,j}$, $t_{11} = t_{33}$ and $t_{12} = \widetilde{t}_{23}$.

Lemma 2.2 *Let T be an upper triangular matrix, nonsingular with non-zero diagonal. Then $T^{-1} = uT\left(\mu_1, ..., \mu_{2n-p}\right)$ and has the following block decomposition,*

$$T^{-1} = \begin{pmatrix} \left(T^{-1}\right)_{11} & \left(T^{-1}\right)_{12} & \left(T^{-1}\right)_{13} \\ 0 & \left(T^{-1}\right)_{22} & \left(T^{-1}\right)_{23} \\ 0 & 0 & \left(T^{-1}\right)_{33} \end{pmatrix} = \begin{pmatrix} t_{11}^{-1} & \widetilde{P} & Q \\ 0 & t_{22}^{-1} & P \\ 0 & 0 & t_{11}^{-1} \end{pmatrix}, \tag{7}$$

where

$$P = T(\mu_2, ..., \mu_n; p; n-p), \quad \widetilde{P} = J_{n-p} P^t J_r,$$

$$t_{22} P + t_{23} t_{11}^{-1} = 0_{(p,n-p)}, \quad h_{11} P + M t_{11}^{-1} = 0_{(p,n-p)}$$

$$t_{11} \widetilde{P} + \widetilde{t}_{23} t_{22}^{-1} = 0_{(n-p,p)}, \quad t_{11} Q + \widetilde{t}_{23} P + t_{13} t_{11}^{-1} = 0_{(n-p,n-p)}.$$

3 Approximate block diagonalization via upper triangular Toeplitz matrices

Here we propose the main results which lead to our approach.

3.1 Approximate reduction for $n \times n$ real Hankel matrix via upper triangular Toeplitz matrices

Theorem 3.1 *Let* $h = H(\varepsilon_1, ..., \varepsilon_{p-1}, h_p, ..., h_{2n-1})$, *where* $\varepsilon_j \in V(0, \mu)$ *for* $j = 1, 2, ..., p-1$ *with* $h_p \notin V(0, \mu)$ *and*

$$T = uT(h_p, ..., h_{2n-1}), \quad T^{-1} = uT(\mu_1, ..., \mu_{2n-p}),$$

$$t = uT(h_p, ..., h_{n+p-1}), \quad t^{-1} = uT(\mu_1, ..., \mu_n).$$

Then

$$h' = \left(t^{-1}\right)^t h t^{-1} = \begin{pmatrix} h'_{11} & \epsilon' \\ (\epsilon')^t & h'_{22} \end{pmatrix}, \tag{8}$$

where

$$h'_{11} = lH(\mu_1, ..., \mu_p) + \left(t_{22}^{-1}\right)^t J_p \Sigma_p t_{22}^{-1}, \tag{9}$$

$$h'_{22} = -H(\mu_{p+2}, ..., \mu_{2n-p}) + P^t J_p \Sigma_p P, \tag{10}$$

$$\epsilon' = \left(t_{22}^{-1}\right)^t J_p \Sigma_p P. \tag{11}$$

Proof Observe that the matrix t is a submatrix of the matrix T because

$$T = \begin{pmatrix} t_{11} & \widetilde{t}_{23} & t_{13} \\ 0 & & t \end{pmatrix} \text{ with } t = \begin{pmatrix} t_{22} & t_{23} \\ 0 & t_{11} \end{pmatrix}.$$

Then,

$$\begin{aligned} \left(t^{-1}\right)^t h t^{-1} &= \begin{pmatrix} \left(t_{22}^{-1}\right)^t & 0 \\ P^t & \left(t_{11}^{-1}\right)^t \end{pmatrix} \begin{pmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{pmatrix} \begin{pmatrix} t_{22}^{-1} & P \\ 0 & t_{11}^{-1} \end{pmatrix} \\ &= \begin{pmatrix} \left(t_{22}^{-1}\right)^t & 0 \\ P^t & \left(t_{11}^{-1}\right)^t \end{pmatrix} \begin{pmatrix} h_{11} t_{22}^{-1} & h_{11} P + h_{12} t_{11}^{-1} \\ h_{21} t_{22}^{-1} & h_{21} P + h_{22} t_{11}^{-1} \end{pmatrix} \end{aligned}$$

and since $TT^{-1} = I$ and $T = J\mathcal{H}$, we have:

- $h_{11}t_{22}^{-1} = J_p\left(J_p h_{11}\right)t_{22}^{-1} = J_p\left(\Sigma_p + t_{22}\right)t_{22}^{-1} = J_p\Sigma_p t_{22}^{-1} + J_p$,
- $h_{11}P + h_{12}t_{11}^{-1} = J\left((Jh_{11})\,P + Jh_{12}t_{11}^{-1}\right) = J\left(\left(\Sigma_p + t_{22}\right)P + Jh_{12}t_{11}^{-1}\right) = J\left(\Sigma_p P\right) + J\left(t_{22}P + Jh_{12}t_{11}^{-1}\right) = J_p\Sigma_p P$,
- $h_{21}t_{22}^{-1} = J\left(Jh_{21}\right)t_{22}^{-1} = J\widetilde{t}_{23}t_{22}^{-1} = -Jt_{11}\widetilde{P}$,
- $h_{21}P + h_{22}t_{11}^{-1} = J\left(Jh_{21}P + Jh_{22}t_{11}^{-1}\right) = J\left(\widetilde{t}_{23}P + t_{13}t_{11}^{-1}\right) = -Jt_{11}Q$.

So,

$$\left(t^{-1}\right)^t ht^{-1} = \begin{pmatrix} \left(t_{22}^{-1}\right)^t & 0 \\ P^t & \left(t_{11}^{-1}\right)^t \end{pmatrix}\begin{pmatrix} J_p\Sigma_p t_{22}^{-1} + J_p & J_p\Sigma_p P \\ -Jt_{11}\widetilde{P} & -Jt_{11}Q \end{pmatrix}$$

$$= \begin{pmatrix} \left(t_{22}^{-1}\right)^t\left(J_p + J_p\Sigma_p t_{22}^{-1}\right) & \left(t_{22}^{-1}\right)^t J_p\Sigma_p P \\ P^t\left(J_p\Sigma_p t_{22}^{-1} + J_p\right) - \left(t_{11}^{-1}\right)^t Jt_{11}\widetilde{P} & P^t J_p\Sigma_p P - \left(t_{11}^{-1}\right)^t Jt_{11}Q \end{pmatrix}.$$

Since

- $\left(t_{22}^{-1}\right)^t\left(J_p + J_p\Sigma_p t_{22}^{-1}\right) = \left(t_{22}^{-1}\right)^t J_p + \left(t_{22}^{-1}\right)^t J_p\Sigma_p t_{22}^{-1} = \left(t_{22}^{-1}\right)^t J_p + \left(t_{22}^{-1}\right)^t J_p\Sigma_p t_{22}^{-1}$,
- $P^t\left(J_p\Sigma_p t_{22}^{-1} + J_p\right) - \left(t_{11}^{-1}\right)^t Jt_{11}\widetilde{P} = P^t J_p - \left(t_{11}^{-1}\right)^t Jt_{11}\widetilde{P} + P^t J_p\Sigma_p t_{22}^{-1} = \left(P^t J_p - Jt_{11}^{-1}t_{11}JP^tJ\right) + P^t J_p\Sigma_p t_{22}^{-1} = P^t J_p\Sigma_p t_{22}^{-1}$,
- $P^t J_p\Sigma_p P - \left(t_{11}^{-1}\right)^t Jt_{11}Q = P^t J_p\Sigma_p P - JQ$,

it follows that

$$\left(t^{-1}\right)^t ht^{-1} = \begin{pmatrix} \left(t_{22}^{-1}\right)^t J_p + \left(t_{22}^{-1}\right)^t J_p\Sigma_p t_{22}^{-1} & \left(t_{22}^{-1}\right)^t J_p\Sigma_p P \\ P^t J_p\Sigma_p t_{22}^{-1} & -JQ + P^t J_p\Sigma_p P \end{pmatrix}.$$

Moreover,

$$t_{22}^{-1} = uT\left(\mu_1, ..., \mu_p\right), \quad Q = T\left(\mu_{p+2}, ..., \mu_{2n-p}\right).$$

Hence, multiplying by J involves

$$\left(t_{22}^{-1}\right)^t J_p = uH\left(\mu_1, ..., \mu_p\right) \text{ and } -JQ = -H\left(\mu_{p+2}, ..., \mu_{2n-p}\right).$$

□

Then, we can give the fast algorithm to compute the reduction (8) as follows:

Algorithm 3.1 (Approximate reduction via Toeplitz for Hankel matrix) Given a list $S = [\varepsilon_1, ..., \varepsilon_{p-1}, h_p, ..., h_{2n-1}]$ where $\varepsilon_j \in V(0, \mu)$ for $j = 1, 2, ..., p-1$ with $h_p \notin V(0, \mu)$, this algorithm computes the reduction of Hankel matrix via upper triangular Toeplitz matrix.

1. Define a Hankel matrix $h = H\left(\varepsilon_1, ..., \varepsilon_{p-1}, h_p, ..., h_{2n-1}\right)$
2. Define an upper triangular Toeplitz matrix $t = uT\left(h_p, ..., h_{n+p-1}\right)$
3. Compute t^{-1}via Matlab inversion
4. Compute $h' = \left(t^{-1}\right)^t ht^{-1}$
5. Set $h'_{11} = h'(1:p, 1:p)$ and $h'_{22} = h'(p+1:n, p+1:n)$.

Example 3.1 Consider the following list S of length 17 which defines a Hankel matrix

$$S = \left[\varepsilon_1 = 1.0e\text{-}13, \frac{1}{3}, -\frac{1}{3}, -\frac{1}{9}, \frac{5}{9}, -\frac{11}{27}, -\frac{4}{9}, \frac{89}{81}, \frac{7}{81}, -\frac{584}{243}, -\frac{104}{243}, \frac{12667}{1458}, \frac{1018}{243}, -\frac{193345}{4374}, -\frac{36133}{2187}, \frac{3042241}{13122}, \frac{853193}{26244}\right].$$

Then, Theorem 3.1 gives the following result:

$$t^{-1} = uT\,(3, 3, 4, -6.6e\text{-}16, -1.3e\text{-}15, 1, -2.3e\text{-}15, -5, 1)\,.$$

Thus,

$$h' = \left(\begin{array}{|cc|cccccccc|} \hline \mathbf{9e\text{-}13} & \mathbf{3} & 1.2e\text{-}12 & 0 & -8.8e\text{-}16 & 3e\text{-}13 & 0 & -1.5e\text{-}12 & 3e\text{-}13 \\ \mathbf{3} & \mathbf{3} & 1.2e\text{-}12 & -8.8e\text{-}16 & -8.8e\text{-}16 & 3e\text{-}13 & 0 & -1.5e\text{-}12 & 3e\text{-}13 \\ \hline 1.2e\text{-}12 & 1.2e\text{-}12 & 1.6e\text{-}12 & 1.1e\text{-}15 & -1 & 4e\text{-}13 & 5 & -1 & -25 \\ -1.6e\text{-}16 & -1.6e\text{-}16 & 1.1e\text{-}15 & -1 & 1.8e\text{-}15 & 5 & -1 & -25 & 21/2 \\ 0 & 0 & -1 & 2.7e\text{-}15 & 5 & -1 & -25 & 21/2 & 124 \\ 3e\text{-}13 & 3e\text{-}13 & 4e\text{-}13 & 5 & -1 & -25 & 21/2 & 124 & -81 \\ 0 & 0 & 5 & -1 & -25 & 21/2 & 124 & -81 & -609 \\ -1.5e\text{-}12 & -1.5e\text{-}12 & -1 & -25 & 21/2 & 124 & -81 & -609 & 545 \\ 3e\text{-}13 & 3e\text{-}13 & -25 & 21/2 & 124 & -81 & -609 & 545 & 11873/4 \\ \hline \end{array}\right).$$

Observe that our result converge to the exact case ($\varepsilon_1 = 0$)

$$h' \simeq \begin{pmatrix} lH\,(0, 3, 3) & 0 \\ 0 & -H\left(0, 0, 1, 0, -5, 1, 25, -\frac{21}{2}, -124, 81, 609, -545, -\frac{11873}{4}\right) \end{pmatrix}.$$

One solution to iterate Theorem 3.1 is to construct a new Hankel matrix with the first column and the last row of h'_{22}. So we have

$$h'_{22} = -H\left(-1.6e\text{-}12, -1.1e\text{-}15, 1, -4e\text{-}13, -5, 1, 25, -\frac{21}{2}, -124, 81, 609, -545, -\frac{11873}{4}\right).$$

This example uses general (Matlab) inversion for an upper triangular Toeplitz matrix t. In the next section, we propose computing this inversion via superfast techniques for obtaining our main algorithm in $O\left(n^2\right)$ arithmetic operations.

Theorem 3.1 gives an approximate reduction for real Hankel matrix. Thus if we have $\varepsilon_j = 0$, Theorem 3.1 provides the exact case. To iterate this result, h'_{22} must be a Hankel matrix, but $-H\left(\mu_{r+2}, ..., \mu_{2p-r}\right)$ is a Hankel matrix and $P^t J_p \Sigma_p P$ is only a symmetric matrix. If we choose ε_j very close to zero, we can

conclude that $\Sigma_p \approx 0_p$, and the process converges to the exact case. To solve this problem, we can choose

$$h'_{22} = -H\left(\mu_{p+2}, ..., \mu_{2n-p}\right) + \Theta \tag{12}$$

where Θ is a Hankel matrix built with the first column and the last row of $P^t J_p \Sigma_p P$.

3.2 Revised Bini inversion technique for triangular Toeplitz matrices

The key of our approach is to be able to compute rapidly and efficiently the inverse of an upper triangular Toeplitz matrix.

In this subsection we describe the correct balance between accuracy and speed in the inversion methods for triangular Toeplitz matrices via superfast techniques. There are two kind of inversion methods of triangular Toeplitz matrices: exact inversion (see for instance [5, 8]) and approximate inversion (see for instance [3, 16] and recently [13]).

To compute quickly and efficiently the inversion of an upper (or lower) triangular Toeplitz matrices

$$T_n = uT\left(t_0, t_1, \ldots, t_{n-1}\right), \tag{13}$$

Bini [3] proposed an approximate method. Let $Z_n = \left[z_{jk}\right]_{j,k=1}^{n}$, where all entries of Z_n are zero except $z_{j+1,j} = 1$ for $j = 1, 2, ..., n-1$. We see that

$$T_n = \sum_{j=0}^{n-1} t_j \left(Z_n^j\right)^t. \tag{14}$$

The idea is to approximate Z_n by $Z_n^{(\varepsilon)} = \left[z_{jk}^{(\varepsilon)}\right]_{j,k=1}^{n}$, where $z_{jk}^{(\varepsilon)} = z_{jk}$ when $(j,k) \neq (1,n)$ and $z_{1n}^{(\varepsilon)} = \varepsilon^n$ (ε is a small positive number) and to compute efficiently, by using fast Fourier Transforms, the inverse of

$$T_n^{(\varepsilon)} = \sum_{j=0}^{n-1} t_j \left(\left(Z_n^{(\varepsilon)}\right)^j\right)^t. \tag{15}$$

The inverse of $T_n^{(\varepsilon)}$ can be computed fast by using the Fourier diagonalization

$$\left(T_n^{(\varepsilon)}\right)^{-1} = \left(D_n^{(\varepsilon)}\right)^{-1} F_n^* D_n^{-1} F_n D_n^{(\varepsilon)} \tag{16}$$

where $D_n^{(\varepsilon)} = diag(1, \varepsilon, ..., \epsilon^{n-1})$, $D_n = diag\,(d)$ with $d = \sqrt{n} F_n D_n^{(\varepsilon)} \left[t_j\right]_{j=0}^{n-1}$ and F_n is $n \times n$ Fourier matrix. Equation (16) and [3] conclude the Fast Bini's inversion for triangular Toeplitz matrix.

Recently, Lin et al. [13] propose to revise Bini's algorithm embedding the $n \times n$ triangular Toeplitz matrix into an $2n \times 2n$ triangular Toeplitz matrix.

The revised algorithm can be applied to obtain a faster and more accurate approximate inversion than any superfast (approximate or exact) inversion. The algorithm can be formulated as follows:

Algorithm 3.2 (Fast Revised Bini's inversion for triangular Toeplitz matrix) Given the last column $\left[t_j\right]_{j=0}^{n-1}$ of the upper triangular Toeplitz matrix T_n and choose $\varepsilon \in (0; 1)$, this algorithm computes the last column $\left[\mu_j\right]_{j=0}^{n-1}$ of $(T_n^{(\varepsilon)})^{-1}$.

1. Compute $\widetilde{t}_j = t_j \varepsilon^j$ for $j = 0, 1, ..., n-1$
2. Set $\widetilde{t}_j = 0$ for $j = n, n+1, \ldots, 2n-1$
3. Compute $b = (\sqrt{n} F_{2n}) \left[\widetilde{t}_j\right]_{j=0}^{2n-1}$
4. Compute $d = \left[d_j\right]_{j=0}^{2n-1} = \left[1/b_j\right]_{j=0}^{2n-1}$
5. Compute $f = (F_{2n}^*/\sqrt{2n})d$
6. Compute $\left[\mu_j\right]_{j=0}^{n-1} = \left[f_j/\varepsilon^j\right]_{j=0}^{n-1}$.

Remark 3.1 The best superfast exact inversion technique of the triangular Toeplitz matrices based on divide-and-conquer method requires about 10 FFT(n). However, the approximate method based on Bini inversion of the triangular Toeplitz matrices is the fastest (2 FFT(n)) but is not accurate when the size of the matrices grows. That is why our choice of the Revised Bini inversion is not arbitrary because this effective approach seems to be the fastest (2 FFT(2n)) [13]. Numerical tests show stability and the accuracy of the proposed approach (see Section 5).

Then, by Theorem 3.1 and (12), we can give the fast algorithm to compute the reduction (8) as follows:

Algorithm 3.3 (Approximate reduction via revised Bini's inversion for Hankel matrix) Given a list $S = [\varepsilon_1, ..., \varepsilon_{p-1}, h_p, ..., h_{2n-1}]$ where $\varepsilon_j \in V(0, \mu)$ for $j = 1, 2, ..., p-1$ with $h_p \notin V(0, \mu)$, this algorithm computes the reduction of Hankel matrix via upper triangular Toeplitz matrix.

1. Define a Hankel matrix $h = H\left(\varepsilon_1, ..., \varepsilon_{p-1}, h_p, ..., h_{2n-1}\right)$
2. Define an upper triangular Toeplitz matrix $t = uT\left(h_p, ..., h_{n+p-1}\right)$
3. Call Algorithm 3.2 to calculate t^{-1}
4. Compute $h' = \left(t^{-1}\right)^t h t^{-1}$
5. Set $h'_{11} = h'(1:p, 1:p)$ and $h'_{22} = h'(p+1:n, p+1:n)$.

If we iterate the step of Algorithm 3.1, with an approximate Hankel matrix h'_{22} defined as (12), we will arrive at the approximate block diagonal matrix D_ε as defined in (2) where every block is an approximate lower Hankel matrix and we can give the following algorithm:

Algorithm 3.4 (Approximate block diagonalization for Hankel matrix) Given a list $S = [\varepsilon_1, ..., \varepsilon_{p-1}, h_p, ..., h_{2n-1}]$ where $\varepsilon_j \in V(0, \mu)$ for $j = 1, 2, ..., p-1$

with $h_p \notin V(0, \mu)$, this algorithm computes an approximate block diagonalization for Hankel matrix.

1. Define a Hankel matrix $h = H\left(\varepsilon_1, ..., \varepsilon_{p-1}, h_p, ..., h_{2n-1}\right)$
2. While $(length(S) \geq 2)$ do
   Call Algorithm 3.1
   If $(n > p)$
   Set $S = [h'(p+1, p+1:n)\ h'(n, p+2:n)]$
   EndIf
   EndWhile

4 Approximate block diagonalization via Schur complementation

The classical Schur algorithm is one of the best known techniques to compute

$$h = LDL^t \tag{17}$$

which gives the decomposition of Hankel matrix in $O\left(n^2\right)$ operations (see [5, 6] and [4]). In this section we outline this factorization in an approximate approach. We also clarify the correlation between the classical approximate factorization method and the approximate diagonalization obtained with our approach.

4.1 Approximate reduction for $n \times n$ real Hankel matrix via Schur complementation

Before starting the approximate factorization of Schur complementation algorithm for Hankel matrix, we propose the following theorem.

Theorem 4.1 *Let h be a real symmetric matrix satisfying the following partition*

$$h = \begin{pmatrix} h_{11} & h_{21}^t \\ h_{21} & h_{22} \end{pmatrix} \tag{18}$$

where

$$h_{11} = lH\left(h_1, ..., h_p\right) + J_p \Sigma_p,\ h_{21} \in \mathbb{R}^{(n-p)\times p} \text{ and } h_{22} \in \mathbb{R}^{(n-p)\times(n-p)},$$

consider the Schur complement of h_{11}

$$h_{sc} = h_{22} - h_{21}\left(lH\left(h_1, ..., h_p\right)\right)^{-1} h_{21}^t$$

and the elimination matrix

$$l = \begin{pmatrix} I_{p\times p} & 0_{p\times(n-p)} \\ h_{21}\left(lH\left(h_1, ..., h_p\right)\right)^{-1} & I_{(n-p)\times(n-p)} \end{pmatrix}.$$

Then

$$lh'l^t = h \ \ where \ \ h' = \begin{pmatrix} h'_{11} & \epsilon' \\ \left(\epsilon'\right)^t & h'_{sc} \end{pmatrix} \tag{19}$$

where

$$h'_{11} = h_{11},\ \epsilon' = -J_p \Sigma_p \left(h_{21} \left(lH\left(h_1, ..., h_p\right)\right)^{-1}\right)^t,$$

$$h'_{sc} = h_{sc} + \left(h_{21} \left(lH\left(h_1, ..., h_p\right)\right)^{-1}\right) J_p \Sigma_p \left(h_{21} \left(lH\left(h_1, ..., h_p\right)\right)^{-1}\right)^t.$$

Proof We will show that

$$h = lh'l^t.$$

Then h' can be multiplied as

$$\begin{aligned} lh' &= \begin{pmatrix} I_{p\times p} & 0_{p\times(n-p)} \\ h_{21}\left(lH\left(h_1, ..., h_p\right)\right)^{-1} & I_{(n-p)\times(n-p)} \end{pmatrix} \begin{pmatrix} lH\left(h_1, ..., h_p\right) + J_p\Sigma_p & \epsilon' \\ \left(\epsilon'\right)^t & h'_{sc} \end{pmatrix} \\ &= \begin{pmatrix} lH\left(h_1, ..., h_p\right) + J_p\Sigma_p & \epsilon' \\ h_{21} + h_{21}\left(lH\left(h_1, ..., h_p\right)\right)^{-1} J_p\Sigma_p + \left(\epsilon'\right)^t h_{21}\left(lH\left(h_1, ..., h_p\right)\right)^{-1}\epsilon' + h'_{sc} \end{pmatrix} \\ &= \begin{pmatrix} lH\left(h_1, ..., h_p\right) + J_p\Sigma_p & \epsilon' \\ h_{21} & h_{21}\left(lH\left(h_1, ..., h_p\right)\right)^{-1}\epsilon' + h'_{sc} \end{pmatrix} \\ &= \begin{pmatrix} lH\left(h_1, ..., h_p\right) + J_p\Sigma_p & \epsilon' \\ h_{21} & h_{sc} \end{pmatrix}. \end{aligned}$$

Thus, lh' can be multiplied as

$$\begin{aligned} \left(lh'\right)l^t &= \begin{pmatrix} h_{11} & \epsilon' \\ h_{21} & h_{sc} \end{pmatrix} \begin{pmatrix} I_{p\times p} & \left(\left(lH\left(h_1, ..., h_p\right)\right)^{-1}\right)^t h_{21}^t \\ 0_{(n-p)\times p} & I_{(n-p)\times(n-p)} \end{pmatrix} \\ &= \begin{pmatrix} h_{11} & h_{21}^t + J_p\Sigma_p\left(h_{21}\left(lH\left(h_1, ..., h_p\right)\right)^{-1}\right)^t h_{21}^t + \epsilon' \\ h_{21} & h_{21}\left(\left(lH\left(h_1, ..., h_p\right)\right)^{-1}\right)^t h_{21}^t + h_{21}\left(lH\left(h_1, ..., h_p\right)\right)^{-1}\epsilon' + h'_{sc} \end{pmatrix} \\ &= \begin{pmatrix} h_{11} & h_{21}^t \\ h_{21} & h_{21}\left(\left(lH\left(h_1, ..., h_p\right)\right)^{-1}\right)^t h_{21}^t + h_{sc} \end{pmatrix}. \end{aligned}$$

□

Example 4.1 Consider the same assumption as defined in Example 3.1. Then, we have

$$h_{11} = \begin{pmatrix} 1e\text{-}13 & \frac{1}{3} \\ \frac{1}{3} & -\frac{1}{3} \end{pmatrix}, h_{12} = h_{21}^t = \begin{pmatrix} -\frac{1}{3} & -\frac{1}{9} & \frac{5}{9} & -\frac{11}{27} & -\frac{4}{9} & \frac{89}{81} & \frac{7}{81} \\ -\frac{1}{9} & \frac{5}{9} & -\frac{11}{27} & -\frac{4}{9} & \frac{89}{81} & \frac{7}{81} & -\frac{584}{243} \end{pmatrix}, \text{ and}$$

$$h_{22} = -H\left(\frac{5}{9}, -\frac{11}{27}, -\frac{4}{9}, \frac{89}{81}, \frac{7}{81}, -\frac{584}{243}, -\frac{104}{243}, \frac{12667}{1458}, \frac{1018}{243}, -\frac{193345}{4374}, -\frac{36133}{2187}, \frac{3042241}{13122}, \frac{853193}{26244}\right).$$

Thus,

$$l = \begin{pmatrix} 1 & 0 & 0\,0\,0\,0\,0\,0\,0 \\ 0 & 1 & 0\,0\,0\,0\,0\,0\,0 \\ -1.3333 & -1 & 1\,0\,0\,0\,0\,0\,0 \\ 1.3333 & -0.33333 & 0\,1\,0\,0\,0\,0\,0 \\ 0.44444 & 1.6667 & 0\,0\,1\,0\,0\,0\,0 \\ -2.5556 & -1.2222 & 0\,0\,0\,1\,0\,0\,0 \\ 1.963 & -1.3333 & 0\,0\,0\,0\,1\,0\,0 \\ 3.5556 & 3.2963 & 0\,0\,0\,0\,0\,1\,0 \\ -6.9506 & 0.25926 & 0\,0\,0\,0\,0\,0\,1 \end{pmatrix}, \quad h'_{11} = h_{11},$$

$$\epsilon' = \begin{pmatrix} 1.3e\text{-}13 & -1.3e\text{-}13 & -4.4e\text{-}14 & 2.6e\text{-}13 & -2e\text{-}13 & -3.6e\text{-}13 & 7e\text{-}13 \\ -1.1e\text{-}16 & 2.3e\text{-}17 & 4.4e\text{-}16 & 2.8e\text{-}16 & -2.7e\text{-}15 & -7.5e\text{-}16 & 1.3e\text{-}14 \end{pmatrix},$$

$$h'_{sc} = \begin{pmatrix} \mathbf{1.8e\text{-}13} & \mathbf{-1.8e\text{-}13} & \mathbf{-0.11111} & 0.11111 & 0.59259 & -0.85185 & -2.716 \\ \mathbf{-1.8e\text{-}13} & \mathbf{-0.11111} & \mathbf{0.22222} & 0.48148 & -1.4444 & -1.8642 & 7.7716 \\ \mathbf{-0.11111} & \mathbf{0.22222} & \mathbf{0.51852} & -1.4815 & -2.0617 & 8.0556 & 8.1564 \\ 0.11111 & 0.48148 & -1.4815 & -2.0123 & 8.8951 & 7.1029 & -46.92 \\ 0.59259 & -1.4444 & -2.0617 & 8.8951 & 6.5267 & -46.245 & -19.896 \\ -0.85185 & -1.8642 & 8.0556 & 7.1029 & -46.245 & -20.713 & 239.46 \\ -2.716 & 7.7716 & 8.1564 & -46.92 & -19.896 & 239.46 & 33.734 \end{pmatrix}.$$

This example uses a general (Matlab) inversion for an upper triangular Hankel matrix $lH\left(h_1, ..., h_p\right)$. In the following, we propose to compute an approximate reduction for symmetric matrix as defined in (18) via Schur Complementation and Revised Bini's inversion in $O\left(n^2\right)$ arithmetic operations. Indeed we can easily write $\left(lH\left(h_1, ..., h_p\right)\right)^{-1} = \left(uT\left(h_1, ..., h_p\right)\right)^{-1} J_p$. Thus, we are led to the following algorithm:

Algorithm 4.1 (Approximate reduction via Schur Complementation for symmetric matrix as defined in (18)) Given a symmetric matrix h as defined in (18), this algorithm computes the reduction of h via Schur Complementation.

1. Define a symmetric matrix h as defined in (18)
2. Set $lh_{11} = lH\left(h_1, ..., h_p\right)$ and $ut_{11} = uT\left(h_1, ..., h_p\right)$
3. Call Algorithm 3.2 to calculate $(ut_{11})^{-1}$

4. Compute $(lh_{11})^{-1} = (ut_{11})^{-1} J_p$
5. Set $h_{11} = h(1:p, 1:p)$, $h_{21} = h(p+1:n, 1:p)$ and $h_{22} = h(p+1:n, p+1:n)$
6. Compute $h_{sc} = h_{22} - h_{21}(lh_{11})^{-1} h_{21}^t$
7. Set $l = [I(p,p), h_{21}(lh_{11})^{-1}; 0(n-p,p), I(n-p,n-p)]$
8. Set $invl = [I(p,p), -h_{21}(lh_{11})^{-1}; 0(n-p,p), I(n-p,n-p)]$
9. Compute $h' = (invl)\, h\, (invl)^t$
10. Set $h'_{11} = h'(1:p, 1:p)$ and $h'_{sc} = h'(p+1:n, p+1:n)$.

Remark 4.1

1. To conclude the exact reduction with Algorithm 4.1, we should take $\varepsilon_j = 0$. In this case, $h'_{sc} = h_{sc}$ and H is a symmetric matrix if h_{11} and h_{sc} are also symmetric. Thus, we can iterate this procedure until arriving at the classical diagonalization for a symmetric matrix.
2. Always in the exact case, if h is a Hankel matrix, then it turns out that some iterations of the Schur complementation process (Algorithm 3.4) give us the special form proposed above (17), that is, the blocks of the matrix D are lower triangular Hankel matrices and the L matrix is block lower triangular. This may be somewhat surprising because the Schur complement h_{sc} in every iteration is not a Hankel matrix, but its smallest nonsingular leading principal submatrix turns out to be lower triangular Hankel. (see [4], chapter 2).

If we iterate Algorithm 4.1 with a Hankel matrix H as defined in (1), we will arrive at the approximate block diagonal matrix D_ε as defined in (2) where every block is an approximate lower Hankel matrix and we can give the following fast algorithm:

Algorithm 4.2 (Approximate block factorization via Schur Complementation for Hankel matrix) Given a list $S = [\varepsilon_1, ..., \varepsilon_{p-1}, h_p, ..., h_{2n-1}]$ where $\varepsilon_j \in V(0,\mu)$ for $j = 1, 2, ..., p-1$ with $h_p \notin V(0,\mu)$, this algorithm computes an approximate block diagonalization for Hankel matrix.

1. Define a Hankel matrix $h = H(\varepsilon_1, ..., \varepsilon_{p-1}, h_p, ..., h_{2n-1})$
2. While $(length(S) \geq 2)$ do
   Call Algorithm 4.1
   EndWhile

4.2 Comparison with our algorithm

In this subsection, we explicitly compare our processus with the classical approximate block diagonalization of Hankel matrix (Algorithm 4.2). To clarify the differences and the similarities between these two approaches, we show the results of the various steps by an example. Here, we choose the same assumption as for Example 3.1.

Figures 1 and 2 describe the steps executed with Algorithm 3.4 and Algorithm 4.2 respectively. In Fig. 3, we propose the block diagonal matrices as well as the upper triangular matrices, as defined in (2), obtained with the two approximate methods.

Then, numerical examples reveal the following conclusions:

- Through the steps of the two approaches, we keep the same dimension and the structure of lower Hankel matrix of the diagonal block.

```
Step 1:

D1 =

        9e-013            3
             3            3

H22 =

   1.6011e-012  1.3323e-015           -1  -6.0704e-012            5           -1
   6.6613e-016           -1 -1.2942e-011             5           -1          -25
            -1 -1.2943e-011             5            -1          -25         10.5
  -6.0698e-012            5            -1          -25         10.5          124
             5           -1          -25          10.5          124          -81
            -1          -25         10.5           124          -81         -609
           -25         10.5          124           -81         -609          545

Step 2:

D2 =

   1.6011e-012  1.3323e-015           -1
   1.3323e-015           -1 -6.0292e-010
  -1           -6.0292e-010           -5

H22 =

   3.3125e-012          0.5  5.4729e-009            -1
   0.5           5.626e-009           -1   2.6364e-008
   5.4729e-009           -1  2.6359e-008          -1.5
  -1            2.6364e-008         -1.5          10.5

Step 3:

D3 =

    1.325e-011            2
             2 -2.1892e-008

H22 =

   1.9308e-007          -14
           -14           42

Step 4:

Set

D4=

   1.9308e-007          -14
           -14           42

End STOP.
```

Fig. 1 Describes the steps executed with Algorithm 3.4

```
Step 1:
D1 =

             1e-013       0.33333
           0.33333      -0.33333
Sc1 =

      -1.9784e-013 -6.5892e-014    -0.11111      0.11111      0.59259     -0.85185       -2.716
      -6.5892e-014     -0.11111     0.22222      0.48148      -1.4444      -1.8642       7.7716
          -0.11111      0.22222     0.51852      -1.4815      -2.0617       8.0556       8.1564
           0.11111      0.48148     -1.4815      -2.0123       8.8951       7.1029       -46.92
           0.59259      -1.4444     -2.0617       8.8951       6.5267      -46.245      -19.896
          -0.85185      -1.8642      8.0556       7.1029      -46.245      -20.713       239.46
            -2.716       7.7716      8.1564       -46.92      -19.896       239.46       33.734
Step 2:
D2 =

      -1.9784e-013 -6.5892e-014    -0.11111
      -6.5892e-014     -0.11111     0.22222
          -0.11111      0.22222     0.51852
Sc2 =

       1.3899e-011     0.055556    -0.055556     -0.40741
          0.055556     -0.11111    -0.35185      0.94444
         -0.055556     -0.35185     0.81481       2.0617
          -0.40741      0.94444      2.0617      -5.9321
Step 3:
D3 =

       1.3899e-011     0.055556
          0.055556     -0.11111
Sc3 =

      -8.5885e-010     -0.38889
          -0.38889       1.9444
Step 4:
D4 =

      -8.5885e-010     -0.38889
          -0.38889       1.9444
```

Fig. 2 Describes the steps executed with Algorithm 4.2

- In the first step, the block diagonal h'_{11} as defined in (8) and (19) is the first principal leading submatrix of h whose determinant is non-zero and the inverse of the first principal leading submatrix h whose determinant is non-zero, respectively.
- The upper triangular matrices, as defined in (2) for our approach is the product of Toeplitz-like matrices. However for the complement Schur approach it has not a special structure, only the entries on the main diagonal are equal to one.
- The two approaches require the computation of the upper triangular matrices, as defined in (2) to obtain the block diagonal matrices.
- At the intermediate steps, the two approaches compute the reduction of h via only one triangular Toeplitz inversion.
- In [6, 9], we find a result asserting that a Hankel matrix is LU-equivalent to a block diagonal matrix if the upper triangular matrices, as defined in (2), have diagonal entries equal to one. Here, we may introduce the same approach in an approximate way. The upper triangular matrices, as defined in (2) of the Schur Complementation method is an "approximate" LU-equivalent to a block diagonal matrix.

```
ourA =
            3            3           -4 -1.6381e-012          -20            6 -1.0659e-010            8  1.0278e-007
            0            3           -3           -4          -15          -34            6          -71            8
            0            0           -3           -3          -19          -24          -34          -45          -71
            0            0            0           -3           -3          -38          -24         -110          -45
            0            0            0            0           -3           -6          -38          -36         -110
            0            0            0            0            0           -6           -6          -50          -36
            0            0            0            0            0            0           -6           -6          -50
            0            0            0            0            0            0            0           -6           -6
            0            0            0            0            0            0            0            0           -6


Aschur =

            1            0      -1.3333       1.3333       0.44444      -2.5556       1.963       3.5556      -6.9506
            0            1           -1      -0.33333       1.6667      -1.2222      -1.3333       3.2963       0.25926
            0            0            1            0            0           -4      -1.6667         27.5      -1.4444
            0            0            0            1            0      -6.3333       2.3333       32.111      -21.056
            0            0            0            0            1           -1      -5.3333       7.6667       24.444
            0            0            0            0            0            1            0      -8.3333       2.3333
            0            0            0            0            0            0            1           -1      -7.3333
            0            0            0            0            0            0            0            1            0
            0            0            0            0            0            0            0            0            1


ourD =

       9e-013            3  -1.199e-012   1.0214e-014  -6.0334e-012  -1.3545e-012   2.7812e-011   2.2001e-010   1.8433e-009
            3            3  -1.1884e-012  -2.8422e-014  -7.6088e-012   2.6461e-011   2.5171e-010   2.0634e-009   1.1777e-007
  -1.1984e-012   -1.188e-012     1.6002e-012     1.5543e-015            -1  -3.1894e-012  -1.2079e-013   -6.418e-012   3.3623e-011
   9.6589e-015  -2.8977e-014     2.4425e-015            -1  -7.8981e-013  -1.2079e-013  -1.6112e-012   3.3623e-011   9.2884e-010
  -6.0318e-012  -7.6052e-012            -1  -8.0336e-013            -5  -1.7647e-011   3.3158e-011   8.9656e-010  -3.2657e-009
  -1.3496e-012   2.6461e-011  -3.2006e-012  -1.2368e-013  -1.7618e-011     3.8418e-012            2  -1.5824e-010   1.5469e-010
   2.7816e-011    2.517e-010  -1.3345e-013  -1.6123e-012   3.3203e-011            2  -3.2741e-010   1.5487e-010   -4.22e-010
   2.2001e-010   2.0634e-009  -6.3889e-012   3.3629e-011   8.9694e-010  -1.5831e-010   1.5591e-010     8.2989e-012           -14
   1.8434e-009   1.1777e-007   3.3608e-011   9.2885e-010  -3.2656e-009   1.5553e-010  -4.2163e-010           -14            42


Dschur =

       1e-013       0.33333   1.3334e-013  -1.3334e-013  -4.4409e-014  -9.9754e-014   1.0014e-013  -1.3212e-013   1.3106e-013
      0.33333      -0.33333   1.9784e-013   6.5947e-014  -3.2979e-013   1.1212e-012  -1.3194e-012   2.3411e-012  -2.6037e-012
   1.3338e-013   1.9778e-013     1.775e-013    -1.7712e-013      -0.11111  -2.2149e-013  -1.3618e-012  -5.2913e-013  -2.5704e-012
  -1.3335e-013   6.6026e-014    -1.7757e-013       -0.11111       0.22222  -1.9854e-012   1.3456e-013  -5.9903e-012   2.4727e-012
  -4.4368e-014  -3.2987e-013       -0.11111       0.22222       0.51852  -1.4504e-012   4.1767e-012  -1.5881e-012   8.9599e-012
  -9.9951e-014   1.1215e-012  -2.1952e-013  -1.9862e-012  -1.4555e-012     1.0655e-013       0.055556   7.671e-011  -2.0956e-011
   9.9917e-014   -1.319e-012  -1.3626e-012   1.3483e-013   4.1835e-012       0.055556       -0.11111   8.0302e-012   3.4204e-011
  -1.3478e-013   2.3421e-012  -5.2657e-013  -6.0009e-012  -1.6001e-012   7.6692e-011   8.0635e-012     1.742e-013      -0.38889
   1.3438e-013  -2.6064e-012  -2.5753e-012   2.4763e-012   8.9862e-012  -2.0953e-011   3.4203e-011       -0.38889        1.9444
```

Fig. 3 The proposed block diagonal matrices as well as the upper triangular matrices

- Then, we see the same observation as in [2]. In fact, we can make the diagonal entries of the upper triangular matrices, as defined in (2) in our algorithm be one by multiplying by $\mathrm{diag}(((A_\epsilon)_{11})^{-1}, ((A_\epsilon)_{22})^{-1}, \ldots, ((A_\epsilon)_{nn})^{-1})$ where $(A_\epsilon)_{jj}$ are the entries on the main diagonal of A_ϵ as defined in (2). Thus, the fact of having one on the diagonal belongs to a new "approximate" LU-equivalence definition.

5 Numerical examples

In this section, we put in evidence the potential advantages of our approximate block diagonalization with respect to the classical approximate factorization method, in terms of numerical stability and in terms of computational cost.

- All algorithms were implemented in Matlab 7.0.4.287 (R2007A) and run on an Intel(R) Core(TM)2 CPU T5600 laptop with a 1.83GHz processor and 2046Mb of RAM.
- All algorithms could be applied to any randomly generated Hankel matrix of size $n \times n$. In the examples, we introduce a Hankel matrix via a perturbation of another Hankel matrix, whose diagonalization with blocks of various sizes is known.

- $\varepsilon = (1.0 \times 10^{-10})^{\frac{1}{n}}$ and $\varepsilon = (0.5 \times 10^{-8})^{\frac{1}{n}}$ are our choice (the best) for Revised Bini's algorithm of our approximate method and the Schur approximate method are, respectively.
- The ε_j for $j = 1, 2, ..., p-1$, as defined in Theorem 1, are random *ke*-13 where k is taken randomly in $(-1, 1)$.
- The relative accuracy of the Schur approximate method and our approximate method are, respectively:

$$\text{Error}_{\text{Sc}} = \frac{\left\| D_{\text{Sc approximate}} - D_{\text{Sc}} \right\|_1}{\| D_{\text{Sc}} \|_1}, \tag{20}$$

$$\text{Error}_{\text{Our}} = \frac{\left\| D_{\text{Our approximate}} - D_{\text{Our}} \right\|_1}{\| D_{\text{Our}} \|_1}. \tag{21}$$

5.1 Accuracy

We present some results from numerical examples illustrating the accuracy of our algorithm with respect to the Schur approximate algorithm for six different sequences defining Hankel matrices as (4).

Then, we show the relative accuracy of our algorithm (20) and the Schur complementation algorithm (21) respectively. The results of these experiments are given in the following tables:

n	$\| D_{\text{Our}} \|_1$	$\left\| D_{\text{Our approximate}} \right\|_1$	$\text{Error}_{\text{Our}}$
9	56	56.0000000112313	2.00657680947224*e*-10
12	11	10.9999999999914	3.19325731008972*e*-08
25	6	6.00000000000205	6.74401226501677*e*-06
54	22	22.0000000000498	9.01689172590669*e*-05
75	7	7.00000000019678	1.57796758427668*e*-06
100	7	7.00000000077631	5.11368596534692*e*-07

n	$\| D_{\text{Sc}} \|_1$	$\left\| D_{\text{Sc approximate}} \right\|_1$	Error_{Sc}
9	2.33333333333327	2.33333333299368	4.578777833076956*e*-10
12	0.56250000000000	0.56250000000000	5.640961041485247*e*-08
25	0.87695312500000	0.87696422784116	1.258481465590000*e*-03
54	14.86111111111165	14.88540173365934	1.634509180780000*e*-03
75	2.26562500000000	2.26562506215412	6.772119855150810*e*-06
100	0.95312500000000	0.95312500001764	8.816858555558903*e*-05

5.2 Computational cost and time

It is clear that our two methods of block diagonalization require $O(n^2)$ operations. Therefore it proves that it's interesting to know the execution time

necessary for each method. In the following table, we test the computational times (in seconds) of our approximate block diagonalization and the Schur complementation approximate approach for ten different sizes of Hankel matrices.

n	$\text{Time}_{\text{Sc approximate}}$	$\text{Time}_{\text{Our approximate}}$
9	0.015353	0.008857
12	0.010140	0.005759
25	0.008062	0.006055
54	0.017775	0.009857
75	0.019114	0.017255
100	0.034030	0.029891
200	0.102005	0.108442
512	1.296836	1.076396
750	4.356359	3.311806
1024	9.074938	7.480839

Thus, we observe that the Schur factorization require more time than our approximate approach.

5.3 Stability issues

A theoretical study of the stability of our algorithm is not easy. So, we propose to base ourselves on purely numerical results. Thus, all algorithms are designed for finite precision arithmetic, we perturbate the Hankel matrices to know how the (rounding) errors propagate into the output.

We consider here $S = [0, \frac{1}{3}, -\frac{1}{3}, -\frac{1}{9}, \frac{5}{9}, -\frac{11}{27}, -\frac{4}{9}, \frac{89}{81}, \frac{7}{81}, -\frac{584}{243}, -\frac{104}{243}, \frac{12{,}667}{1458}, \frac{1{,}018}{243}, -\frac{193{,}345}{4374}, -\frac{36{,}133}{2187}, \frac{3{,}042{,}241}{13122}, \frac{853{,}193}{26244}]$ and add 10^{-k} to all entries of Hankel matrix with $k = 7, \ldots, 27$. Our exact blocks and the exact blocks via Schur Complementation in the diagonal matrices have been taken from Theorem 3.1 and Theorem 4.1 respectively when $\varepsilon_j = 0$:

$$D_{\text{Our}} = \text{diag}\left(lH\left(0,3,3\right),\ lH\left(0,0,-1,0,-5\right),\ lH\left(0,2,0\right),\ lH\left(0,-14,42\right)\right),$$

$$D_{\text{Sc}} = \text{diag}\left(lH\left(0,\frac{1}{3},\ \frac{1}{3}\right),\ lH\left(0,0,-\frac{1}{9},\ \frac{2}{9},\ \frac{14}{27}\right),\right.$$

$$\left.lH\left(0,\frac{1}{18},\ -\frac{1}{9}\right),\ lH\left(0,-\frac{7}{18},\ \frac{35}{18}\right)\right).$$

In the following table, we note that the two algorithms are stable starting from $k = 8$ (for $k = 1, \ldots, 7$ we lose our block diagonalization).

k	$\text{Error}_{\text{Sc approximate}}$	$\text{Error}_{\text{Our approximate}}$
8	1.2824*e*-005	6.9946*e*-005
9	1.4636*e*-005	7.0289*e*-006
10	1.5323*e*-005	7.2427*e*-007
11	1.5390*e*-005	6.1916*e*-008
12	1.5399*e*-005	2.9813*e*-008
13	1.5399*e*-005	4.2168*e*-008
14	1.5398*e*-005	2.0506*e*-008
15	1.5398*e*-005	2.5140*e*-008
16	1.5398*e*-005	2.2178*e*-008
17, ..., 27	1.5399*e*-005	2.6585*e*-008

An wider stability analysis applied for our sequences, confirms "numerically" that our two approaches are stable and the perturbation above do not diverge the result starting from a certain additive tolerance.

6 Conclusion

We have presented a fast method for an approximate block diagonalization of an $n \times n$ real Hankel matrix. To motivate our approach, we have also designed an approximate factorization variant via Schur complementation adapted to the $n \times n$ real Hankel matrix. Based on the numerical experiments, our algorithms have been shown to be stable, accurate and $O(n^2)$.

Since all algorithms are designed for finite precision arithmetic, finding right (rounding) errors introduced at the intermediate steps and knowing its propagation into the output is not trivial.

Acknowledgements I thank Prof. H. Lombardi who suggested this problem to me, G. M. Diaz-Toca for the helpful discussions, Prof. C. Brezinski and the referees for valuable suggestions which significantly improved the presentation of this paper.

References

1. Akhiezer, N.I., Krein, M.: Some questions in the theory of moments. Amer. Math. Soc. (1962)
2. Ben Atti, N., Diaz-Toca, G.M.: Block diagonalization and LU-equivalence of Hankel matrices. Linear Algebra Appl. **412**, 247–269 (2006)
3. Bini, D.: Parallel solution of certain Toeplitz linear systems. SIAM J. Comput. **13**, 268–276 (1984)

4. Bultheel, A., Van Barel, M.: Linear algebra, rational approximation and orthogonal polynomials. In: Studies in Computational Mathematics, vol. 6. Elsevier/North-Holland, Amsterdam (1997)
5. Bini, D., Pan, V.: Polynomial and matrix computations. Fundamental Algorithms, vol. 1. Birkhäuser (1994)
6. Gragg, W.B., Lindquist, A.: On partial realization problem. Linear Algebra Appl. **50**, 277–319 (1983)
7. Golub, G.H., Milanfar, P., Varah, J.: A stable numerical method for inverting shape from moments. Technical Report SCCM-97-10, Sei. Comp. and Comp'l. Math. Pgm. Stanford University (1997)
8. Golub, G., Van Loan, C.: Matrix Computations. 3ème édition. J. Hopkins Univ. Press, Baltimore and London (1996)
9. Heining, G., Rost, K.: Algebraic Methods for Toeplitz-like Matrices and Operators. Birkhäuser Verlag, Basel (1984)
10. Heinig, G., Jungnickel, U.: Hankel matrices generated by Markov parameters, Hankel matrix extension, partial realization, and Pade approximation. In: Operator Theory: Advances and Applications, vol. 19, pp. 231–254. Birkhauser, Boston (1986)
11. Kalman, R.E.: On partial realizations, transfer functions and canonical forms. Acta Polytech. Stand. **MA31**, 9–32 (1979)
12. Kung, S.Y.: Multivariable and Multidimensional Systems. Ph.D. thesis, Stanford University, Stanford, CA, (June 1977)
13. Lin, F.-R., Ching, W.-K., Ng, M.K.: Fast inversion of triangular Toeplitz matrices. Theor. Comp. Sci. **315**, 511–523 (2004)
14. Pal, D.: Fast algorithms for structured matrices with arbitrary rank profile. Ph.D. thesis, Stanford University, Stanford (1990)
15. Pan, V.Y.: Structured Matrices and Polynomials: Unified Superfast Algorithms. Springer Verlag (2001)
16. Pan, V.Y., Chen, Z.Q.: Approximate real polynomial division via approximate inversion of real triangular Toeplitz matrices. Appl. Math. Lett. **12**, 1–2 (1999)
17. Pal, D., Kailath, T.: Fast triangular factorization of hankel and related matrices with arbitrary rank profile. SIAM J. Matrix Anal. Appl. **16**, 451–478 (1990)
18. Phillips, J.: The triangular decomposition of Hankel matrices. Math. Comput. **25**, 599–602 (1971)
19. Rissanen, J.: Algorithms for triangular decomposition of block Hankel and Toeplitz matrices with applications to factoring positive polynomials. Math. Comput. **27**, 147–154 (1973)