ELSEVIER Linear Algebra and its Applications 282 (1998) 275–295

LINEAR ALGEBRA AND ITS APPLICATIONS

A fast Hankel solver based on an inversion formula for Loewner matrices

Peter Kravanja, Marc Van Barel *,[1]

Department of Computer Science, Katholieke Universiteit Leuven, Celestijnenlaan 200 A, B-3001 Heverlee, Belgium

Received 1 October 1996; accepted 19 April 1998

Submitted by G. Heinig

Abstract

We propose a new $O(n^2)$ algorithm for solving complex $n \times n$ linear systems that have Hankel structure. Via FFTs the Hankel system is transformed into a Loewner system. An inversion formula enables us to calculate the inverse of the Loewner matrix explicitely. The parameters that occur in this inversion formula are calculated by solving two rational interpolation problems on the unit circle. We present an $O(n^2)$ algorithm to solve these interpolation problems. One of the advantages of this algorithm is that it incorporates pivoting. We have implemented our Hankel solver in Fortran 90. Numerical examples are included. They show the effectiveness of our pivoting strategy. © 1998 Elsevier Science Inc. All rights reserved.

AMS classification: 65F05; 47B35; 15A09

1. Introduction

Let $H = H_n := [h_{k+l}]_{k,l=0}^{n-1}$ be a regular $n \times n$ complex Hankel matrix, and let $b \in \mathbb{C}^n$. We consider the problem of computing $x := H^{-1}b$.

The exchange matrix $E = E_n := [\delta_{k,n-1-l}]_{k,l=0}^{n-1}$ transforms the Hankel system $Hx = b$ into the Toeplitz system $T\tilde{x} = b$ where $T := HE$ and $\tilde{x} := Ex$. Several fast algorithms for solving Hankel or Toeplitz systems have been developed.

* Corresponding author. Fax: +32 16 327996; e-mail: marc.vanbarel@cs.kuleuven.ac.be.

[1] This research was partially supported by the Fund for Scientific Research-Flanders (FWO-V), project "Orthogonal systems and their applications", grant #G.0278.97.

PII: S0024-3795(98)10079-4

"Fast" means that the arithmetic complexity is $O(n^2)$ instead of $O(n^3)$ for general purpose algorithms that do not exploit structure, e.g., Gaussian elimination. These fast algorithms compute a triangular factorization of H (or T) or of its inverse in a recursive way. More precisely, in step $k \leqslant n$ a factorization of the leading principal submatrix H_k (or its inverse) of H is constructed based on a factorization of H_{k-1}. These algorithms only work if all these leading principal submatrices (sections) are nonsingular, i.e., if H is strongly regular. To overcome this difficulty, methods have been developed to jump over exactly singular sections. This results in a block triangular factorization of H. Nearly singular sections are handled as nonsingular ones. Hence these methods can be applied in exact arithmetic, but they become unstable in floating point arithmetic when nearly singular sections are involved. We refer the reader to [2,36] and the references cited therein.

To overcome this instability, "look-ahead" algorithms have been developed. They look ahead from one well-conditioned section to the next one and jump over the ill-conditioned sections that lie in between. If there are only a few such ill-conditioned sections, then the jumps can be small and the algorithm stays fast. However, in the case of a Hankel system $Hx = b$ having all sections except the last one singular look-ahead algorithms will require $O(n^3)$ flops. Another disadvantage of these methods is that a look-ahead criterion and condition estimator that is reliable as well as cheap is difficult to construct. One has to balance between a severe look-ahead criterion that might lead to big jumps and a less stringent criterion that might give less accurate results. For Toeplitz matrices several look-ahead algorithms have been designed [8,9,13,12,22–24,33]. Some of these are even "superfast", i.e., their arithmetic complexity is $O(n \log^2 n)$ [22–24]. For Hankel matrices, we refer to [7,14,3]. Look-ahead algorithms for block Toeplitz matrices can be found in [15,16,44] and for generalized Sylvester matrices in [5,6].

Only recently a totally different approach was taken to overcome the possible instabilities of the "classical" algorithms. The Gaussian elimination method uses (partial or complete) pivoting to enhance numerical stability. Unfortunately, pivoting destroys the structure of a Hankel or Toeplitz matrix. However, other classes of structured matrices maintain their structure after pivoting and thus fast as well as numerically stable methods can be designed. In [25], Heinig proposed for the first time to transform structured matrices from one class into another and to use pivoting strategies to enhance the numerical stability. It is shown how Toeplitz matrices can be transformed into Cauchy-like matrices by the discrete Fourier transformation, which is a fast and stable procedure (see [39]). For Toeplitz-plus-Hankel matrices this was done in [26]. Real trigonometric transformations were studied in [19,28,30]. A matrix $M = [m_{kl}]$ is called Cauchy-like if, for certain numbers y_k and z_l, the rank of the matrix $[(y_k - z_l)m_{kl}]$ is small compared to the order of M. Pivoting does not destroy the Cauchy-like structure. For Cauchy-like systems

several fast algorithms exist [31,17,18,25,19,36]. Heinig [27] proposed to transform a Toeplitz system into a paired Vandermonde system, which is then solved as a tangential Lagrange interpolation problem. When two steps of his algorithm are combined, one obtains a block-step algorithm [28]. Our approach is related to Heinig's. They both involve the solution of interpolation problems. For an overview of different transformation techniques and algorithms we refer the reader to [30,29,28,16] and the references cited therein.

Pivoting for structured matrices has a good record of numerical performance in several applications. For example, the so-called Leja ordering is the equivalent of partial pivoting for Vandermonde-related structures, see [41,32]. Rational Leja ordering (i.e., for Cauchy matrices) is discussed in [4]. In the context of interpolation, pivoting is numerically shown to be a very successful technique in [20,35]. Numerical experiments in [19,35] showed the practicality of this approach for shift-invariant structures (such as Toeplitz or Toeplitz plus Hankel matrices). For a review about pivoting on structured matrices, we refer to [21].

The outline of this paper is as follows. In Section 2 we will show how the Hankel system $Hx = b$ can be transformed into a Loewner system $Lx' = b'$ in $O(n \log n)$ flops. An explicit formula for L^{-1} enables us to calculate x' as $L^{-1}b'$. This inversion formula for Loewner matrices is discussed in Section 3. It involves certain parameters that can be computed by solving two rational interpolation problems on the unit circle. In Section 4 we present an $O(n^2)$ algorithm to solve these interpolation problems. We conclude with numerical examples in Section 5.

2. Transformation into a Loewner system

Let $y_1, \ldots, y_n, z_1, \ldots, z_n$ be $2n$ mutually distinct complex numbers, and define $\mathbf{y} := (y_1, \ldots, y_n)$ and $\mathbf{z} := (z_1, \ldots, z_n)$. Let $\mathscr{L}(\mathbf{y}, \mathbf{z})$ be the class of matrices

$$\mathscr{L}(\mathbf{y}, \mathbf{z}) := \left\{ \left[\frac{c_k - d_l}{y_k - z_l} \right]_{k,l=1}^{n} \mid c_1, \ldots, c_n, d_1, \ldots, d_n \in \mathbb{C} \right\}.$$

The elements of $\mathscr{L}(\mathbf{y}, \mathbf{z})$ are called Loewner matrices. They bear the name of Karl Loewner who studied them in the context of rational interpolation and monotone matrix functions [40]

The set $\mathscr{L}(\mathbf{y}, \mathbf{z})$ is a linear space over $\mathbb{C}$, and a subspace of the linear space of all the $n \times n$ complex matrices. Since addition of a constant to all the $2n$ parameters c_k, d_l leads to the same Loewner matrix, its dimension is $2n - 1$. The set of all the $n \times n$ complex Hankel matrices also forms a linear subspace of dimension $2n - 1$. Hankel and Loewner matrices are even more closely related. According to Fiedler [10] every Hankel matrix can be transformed into a

Loewner matrix, and vice versa. In this section we will formulate this theorem and discuss our $O(n \log n)$ implementation of the transformation.

First we have to deal with some preliminaries concerning Vandermonde matrices. Let $t_1, \ldots, t_n$ be n complex numbers and define $\mathbf{t} := (t_1, \ldots, t_n)$. The Vandermonde matrix with nodes $t_1, \ldots, t_n$ is given by

$$V(\mathbf{t}) = V(t_1, \ldots, t_n) := \begin{bmatrix} 1 & t_1 & \cdots & t_1^{n-1} \\ \vdots & \vdots & & \vdots \\ 1 & t_n & \cdots & t_n^{n-1} \end{bmatrix}.$$

Let $f_{\mathbf{t}}(z)$ be the monic polynomial of degree n that has zeros $t_1, \ldots, t_n$,

$$f_{\mathbf{t}}(z) := (z - t_1) \cdots (z - t_n),$$

and define

$$f_{\mathbf{t},j}(z) := \prod_{k \neq j} (z - t_k), \qquad j = 1, \ldots, n.$$

Note that $f_{\mathbf{t},j}(z)$ is a monic polynomial of degree $n - 1$ for $j = 1, \ldots, n$. Define the $n \times n$ matrix $W(\mathbf{t})$ by the equation

$$\begin{bmatrix} f_{\mathbf{t},1}(z) \\ f_{\mathbf{t},2}(z) \\ \vdots \\ f_{\mathbf{t},n}(z) \end{bmatrix} = W(\mathbf{t}) \begin{bmatrix} 1 \\ z \\ \vdots \\ z^{n-1} \end{bmatrix}. \tag{1}$$

This means that the jth row of $W(\mathbf{t})$ contains the coefficients of $f_{\mathbf{t},j}(z)$ when written in terms of the standard monomial basis $(1, z, \ldots, z^{n-1})$. Then

$$W(\mathbf{t})[V(\mathbf{t})]^{\mathrm{T}} = \operatorname{diag}(f_{\mathbf{t},1}(t_1), \ldots, f_{\mathbf{t},n}(t_n)) =: D(\mathbf{t}). \tag{2}$$

The Vandermonde matrix $V(\mathbf{t})$ is regular if and only if its nodes $t_1, \ldots, t_n$ are mutually distinct. In that case Eq. (2) implies that $W(\mathbf{t})$ is regular, and also that

$$[V(\mathbf{t})]^{-1} = [W(\mathbf{t})]^{\mathrm{T}} [D(\mathbf{t})]^{-1}. \tag{3}$$

For an early reference to this formula, see, for example, the book by Kowalewski [37].

Let $V(\mathbf{y}, \mathbf{z})$ be the $2n \times 2n$ Vandermonde matrix with nodes $y_1, \ldots, y_n$ and $z_1, \ldots, z_n$, and similarly for $W(\mathbf{y}, \mathbf{z})$.

Theorem 1. *The matrix $L := W(\mathbf{y})H[W(\mathbf{z})]^{\mathrm{T}}$ is a Loewner matrix in $\mathscr{L}(\mathbf{y}, \mathbf{z})$ whose parameters $c_1, \ldots, c_n, d_1, \ldots, d_n$ are given by (up to an arbitrary additive constant $\xi \in \mathbb{C}$)*

$$\begin{bmatrix} c_1 \\ \vdots \\ c_n \\ d_1 \\ \vdots \\ d_n \end{bmatrix} = W(\mathbf{y},\mathbf{z}) \begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_{2n-2} \\ \xi \end{bmatrix}.$$

Proof. See Fiedler [10], Theorem 12. □

Note that L is regular.

A judicious choice of the points $\mathbf{y}$ and $\mathbf{z}$ enables us to transform the Hankel system $Hx = b$ into a Loewner system $Lx' = b'$ in $O(n \log n)$ flops. Let $\omega := \exp(2\pi i/n)$ and suppose from now on that $y_k = \omega^{k-1}$ for $k = 1, \ldots, n$. That is, let $\mathbf{y} = (1, \omega, \ldots, \omega^{n-1})$. Let $\zeta := \exp(\pi i/n)$ and suppose from now on that $z_k = \zeta y_k$ for $k = 1, \ldots, n$. That is, let $\mathbf{z} = (\zeta, \zeta\omega, \ldots, \zeta\omega^{n-1})$. Note that

$$\mathbf{y} = (1, \zeta^2, \ldots, \zeta^{2n-2}) \quad \text{and} \quad \mathbf{z} = (\zeta, \zeta^3, \ldots, \zeta^{2n-1}).$$

Let Ω_n be the $n \times n$ Fourier matrix,

$$\Omega_n := \frac{1}{\sqrt{n}} V(1, \omega, \ldots, \omega^{n-1}). \tag{4}$$

Matrix-vector products involving Ω_n (Ω_n^H) amount to a(n) (inverse) discrete Fourier transform (DFT) and can be evaluated via the celebrated (inverse) fast Fourier transform (FFT) in $O(n \log n)$ flops. Finally, let $D_{n,\omega}$ and $D_{n,\zeta}$ be the $n \times n$ diagonal matrices

$$D_{n,\omega} := \operatorname{diag}(1, \omega, \ldots, \omega^{n-1}) \quad \text{and} \quad D_{n,\zeta} := (1, \zeta, \ldots, \zeta^{n-1}).$$

Theorem 2. *The solution to the Hankel system $Hx = b$ is given by*

$$x = \frac{1}{\sqrt{n}} \zeta^{n-1} \overline{D_{n,\zeta}} \Omega_n^H \overline{D_{n,\omega}} x',$$

where $x' := L^{-1}b'$ with $b' := \sqrt{n}\, \overline{D_{n,\omega}\Omega_n} b$.

Proof. Theorem 1 implies that $x = H^{-1}b$ is given by $x = [W(\mathbf{z})]^{\mathrm{T}} x'$ where $x' := L^{-1}b'$ with $b' := W(\mathbf{y})b$. Eqs. (2) and (4), and the fact that Ω_n is unitary imply that

$$W(\mathbf{y}) = W(1, \omega, \ldots, \omega^{n-1}) = \frac{1}{\sqrt{n}} D(1, \omega, \ldots, \omega^{n-1}) \overline{\Omega_n}. \tag{5}$$

An easy calculation shows that $D(1, \omega, \ldots, \omega^{n-1}) = n\overline{D_{n,\omega}}$ and thus $b' = W(\mathbf{y})b = \sqrt{n}\, \overline{D_{n,\omega}\Omega_n} b$.

Eq. (3) tells us that

$$[W(\mathbf{z})]^{\mathrm{T}} = [V(\mathbf{z})]^{-1} D(\mathbf{z}).$$

Since $V(\mathbf{z}) = V(1, \omega, \ldots, \omega^{n-1}) D_{n,\zeta}$ and $D(\mathbf{z}) = n\zeta^{n-1}\overline{D_{n,\omega}}$, it follows that

$$\begin{aligned} x &= [W(\mathbf{z})]^{\mathrm{T}} x' = n\zeta^{n-1} D_{n,\zeta}^{-1} [V(1, \omega, \ldots, \omega^{n-1})]^{-1} \overline{D_{n,\omega}} x' \\ &= \sqrt{n}\zeta^{n-1} \overline{D_{n,\zeta}} \Omega_n^H \overline{D_{n,\omega}} x'. \end{aligned}$$

This proves the theorem. □

Let P be the permutation matrix defined by

$$P \begin{bmatrix} y_1 \\ \vdots \\ y_n \\ z_1 \\ \vdots \\ z_n \end{bmatrix} = \begin{bmatrix} 1 \\ \zeta \\ \vdots \\ \zeta^{2n-1} \end{bmatrix},$$

let Ω_{2n} be the $2n \times 2n$ Fourier matrix,

$$\Omega_{2n} := \frac{1}{\sqrt{2n}} V(1, \zeta, \ldots, \zeta^{2n-1}),$$

and let $D_{2n,\zeta} := \operatorname{diag}(1, \zeta, \ldots, \zeta^{2n-1})$.

Theorem 3. *The parameters $c_1, \ldots, c_n, d_1, \ldots, d_n$ of the Loewner matrix L are given by (up to an arbitrary additive constant $\xi \in \mathbb{C}$)*

$$\begin{bmatrix} c_1 \\ \vdots \\ c_n \\ d_1 \\ \vdots \\ d_n \end{bmatrix} = \sqrt{2n} P^{\mathrm{T}} \overline{D_{2n,\zeta} \Omega_{2n}} \begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_{2n-2} \\ \xi \end{bmatrix}.$$

Proof. Eqs. (1) and (5) imply that

$$PW(\mathbf{y}, \mathbf{z}) = W(1, \zeta, \ldots, \zeta^{2n-1}) = \frac{1}{\sqrt{2n}} D(1, \zeta, \ldots, \zeta^{2n-1}) \overline{\Omega_{2n}}.$$

An easy calculation shows that $D(1, \zeta, \ldots, \zeta^{2n-1}) = 2n\overline{D_{2n,\zeta}}$ and thus

$$W(\mathbf{y}, \mathbf{z}) = \sqrt{2n} P^{\mathrm{T}} \overline{D_{2n,\zeta} \Omega_{2n}}.$$

The result then follows immediately from Theorem 1. □

Notes. In the proof of Theorem 2 we have shown that

$$W(\mathbf{y}) = \sqrt{n}\,\overline{D_{n,\omega}\Omega_n} \quad \text{and} \quad [W(\mathbf{z})]^{\mathrm{T}} = \sqrt{n}\zeta^{n-1}\overline{D_{n,\zeta}}\Omega_n^H\overline{D_{n,\omega}}.$$

These formulae imply that $W(\mathbf{y})/\sqrt{n}$ and $[W(\mathbf{z})]^{\mathrm{T}}/\sqrt{n}$ are unitary. Therefore $\|L\|_2 = n\|H\|_2$ and $\|L^{-1}\|_2 = n^{-1}\|H^{-1}\|_2$ and thus $\kappa_2(L) = \kappa_2(H)$. In other words, the spectral condition number of H is left unchanged.

In [47] Vavřín uses the nth roots of unity together with these numbers multiplied by 2 to get a similar fast transformation from a Hankel to a Loewner matrix.

3. An inversion formula for Loewner matrices

Theorem 4. *Let $p_k, u_k, \tilde{p}_k$ and $\tilde{u}_k$ for $k = 1, \ldots, n$ be defined by the equations*

$$[p_1 \quad \cdots \quad p_n]L = [1 \quad \cdots \quad 1],$$
$$[u_1 \quad \cdots \quad u_n]L = [d_1 \quad \cdots \quad d_n],$$
$$L\begin{bmatrix}\tilde{p}_1\\ \vdots \\ \tilde{p}_n\end{bmatrix} = \begin{bmatrix}1\\ \vdots \\ 1\end{bmatrix},$$
$$L\begin{bmatrix}\tilde{u}_1\\ \vdots \\ \tilde{u}_n\end{bmatrix} = \begin{bmatrix}c_1\\ \vdots \\ c_n\end{bmatrix}.$$

Then the inverse of L is given by

$$L^{-1} = \left[\frac{\tilde{u}_k p_l - \tilde{p}_k u_l}{y_l - z_k}\right]_{k,l=1}^{n}. \tag{6}$$

Proof. See van Barel and Vavřín [45], Theorem 2.2. □

A Loewner matrix is a special kind of Cauchy-like matrix. The inversion formula of the previous theorem can therefore be seen as a special case of the formula given in the book of Heinig and Rost [31], p. 161.

The parameters that appear in the inversion formula (6) can be computed by solving the following two (linearized) rational interpolation problems.

Theorem 5. *Define*

$$u(z) := f_{\mathbf{y}}(z) - \sum_{k=1}^{n} u_k f_{\mathbf{y},k}(z),$$

$$v(z) := -\sum_{k=1}^{n} c_k u_k f_{\mathbf{y},k}(z),$$

$$\tilde{u}(z) := f_{\mathbf{z}}(z) - \sum_{k=1}^{n} \tilde{u}_k f_{\mathbf{z},k}(z),$$

$$\tilde{v}(z) := -\sum_{k=1}^{n} d_k \tilde{u}_k f_{\mathbf{z},k}(z).$$

Then the polynomial pair $(v(z), u(z))$ *is the only pair such that*

- $u(z) \in \mathbb{C}[z]$ *and* $\deg u(z) = n$, $v(z) \in \mathbb{C}[z]$ *and* $\deg v(z) < n$, $u(z)$ *is monic,*
- $v(y_k) = c_k u(y_k)$ *and* $v(z_k) = d_k u(z_k)$ *for* $k = 1, \ldots, n$.

The polynomial pair $(\tilde{v}(z), \tilde{u}(z))$ *satisfies exactly the same properties and thus* $v(z) \equiv \tilde{v}(z)$ *and* $u(z) \equiv \tilde{u}(z)$.

Proof. See van Barel and Vavřín [45], Theorem 3.1. Compare with Vavřín [47], Theorem 2.1. □

Theorem 6. *Define*

$$p(z) := \sum_{k=1}^{n} p_k f_{\mathbf{y},k}(z),$$

$$q(z) := f_{\mathbf{y}}(z) + \sum_{k=1}^{n} c_k p_k f_{\mathbf{y},k}(z),$$

$$\tilde{p}(z) := \sum_{k=1}^{n} \tilde{p}_k f_{\mathbf{z},k}(z),$$

$$\tilde{q}(z) := f_{\mathbf{z}}(z) + \sum_{k=1}^{n} d_k \tilde{p}_k f_{\mathbf{z},k}(z).$$

Then the polynomial pair $(q(z), p(z))$ *is the only pair such that*

- $q(z) \in \mathbb{C}[z]$ *and* $\deg q(z) = n$, $p(z) \in \mathbb{C}[z]$ *and* $\deg p(z) < n$, $q(z)$ *is monic,*
- $q(y_k) = c_k p(y_k)$ *and* $q(z_k) = d_k p(z_k)$ *for* $k = 1, \ldots, n$.

The polynomial pair $(\tilde{q}(z), \tilde{p}(z))$ *satisfies exactly the same properties and thus* $q(z) \equiv \tilde{q}(z)$ *and* $p(z) \equiv \tilde{p}(z)$.

Proof. See van Barel and Vavřín [45], Theorem 3.2. □

Note that if $p(y_k)$ and $p(z_k)$, $k = 1, \ldots, n$, are different from zero, then the rational function $q(z)/p(z)$ satisfies the proper rational interpolation conditions

$q(y_k)/p(y_k) = c_k$ and $q(z_k)/p(z_k) = d_k$, $k = 1, \ldots, n$. Similarly, if $u(y_k)$ and $u(z_k)$, $k = 1, \ldots, n$, are different from zero, then the rational function $v(z)/u(z)$ satisfies the proper rational interpolation conditions $v(y_k)/u(y_k) = c_k$ and $v(z_k)/u(z_k) = d_k$, $k = 1, \ldots, n$. The rational functions $q(z)/p(z)$ and $v(z)/u(z)$ are different because their degree structure is different.

In the next section we will present an $O(n^2)$ algorithm that solves these interpolation problems. An easy calculation reveals the following connections with the parameters that appear in the inversion formula (6):

$$\begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix} = -\frac{1}{n} D_{n,\omega} \begin{bmatrix} u(y_1) \\ \vdots \\ u(y_n) \end{bmatrix}, \qquad \begin{bmatrix} p_1 \\ \vdots \\ p_n \end{bmatrix} = \frac{1}{n} D_{n,\omega} \begin{bmatrix} p(y_1) \\ \vdots \\ p(y_n) \end{bmatrix},$$

$$\begin{bmatrix} \tilde{u}_1 \\ \vdots \\ \tilde{u}_n \end{bmatrix} = -\frac{1}{n} \overline{\zeta^{n-1}} D_{n,\omega} \begin{bmatrix} u(z_1) \\ \vdots \\ u(z_n) \end{bmatrix}, \qquad \begin{bmatrix} \tilde{p}_1 \\ \vdots \\ \tilde{p}_n \end{bmatrix} = \frac{1}{n} \overline{\zeta^{n-1}} D_{n,\omega} \begin{bmatrix} p(z_1) \\ \vdots \\ p(z_n) \end{bmatrix}.$$

Once these inversion parameters have been computed, the matrix-vector product $x' = L^{-1}b'$ can be calculated in $O(n \log n)$ flops. Indeed, L^{-1} can be written as

$$\begin{aligned} L^{-1} = {} & \operatorname{diag}(\tilde{p}_1, \ldots, \tilde{p}_n)\, C \operatorname{diag}(u_1, \ldots, u_n) \\ & - \operatorname{diag}(\tilde{u}_1, \ldots, \tilde{u}_n) C \operatorname{diag}(p_1, \ldots, p_n) \end{aligned} \tag{7}$$

if C is given by the Cauchy matrix

$$C := \left[\frac{1}{z_k - y_l} \right]_{k,l=1}^{n}.$$

Proposition 3.2. of [11] for the special case of the roots of unity leads to the following factorization of the Cauchy matrix

$$C = -\frac{n}{2} \Omega_n D_{n,\zeta} \Omega_n^H \overline{D_{n,\omega}}.$$

This implies that the product of C with a vector in $\mathbb{C}^n$ can be evaluated in $O(n \log n)$ flops, and thus, because of Eq. (7), the same holds for the product $L^{-1}b'$.

4. Solving the linearized rational interpolation problems

Let $s_k := y_k$ and $s_{n+k} := z_k$ for $k = 1, \ldots, n$. Define the column vectors $f_1, \ldots, f_{2n} \in \mathbb{C}^{2\times 1}$ as

$$f_k := \begin{bmatrix} 1 \\ -c_k \end{bmatrix}, \qquad f_{n+k} := \begin{bmatrix} 1 \\ -d_k \end{bmatrix}, \quad k = 1, \ldots, n.$$

Consider the interpolation problem

$$f_k^{\mathrm{T}} B(s_k) = [0 \ 0], \quad k = 1, \ldots, 2n, \tag{8}$$

where

$$B(z) = \begin{bmatrix} n_l(z) & n_r(z) \\ d_l(z) & d_r(z) \end{bmatrix} \in \mathbb{C}[z]^{2\times 2}$$

with $\deg n_l(z) = n$, $\deg d_l(z) \leqslant n-1$, $\deg n_r(z) \leqslant n-1$ and $\deg d_r(z) = n$, and $n_l(z)$ as well as $d_r(z)$ monic, i.e., where $B(z)$ is a monic 2×2 matrix polynomial of degree n. Theorems 5 and 6 assert that this problem has a unique solution given by

$$B^\star(z) := \begin{bmatrix} q(z) & v(z) \\ p(z) & u(z) \end{bmatrix}.$$

Algorithm 7 calculates a 2×2 matrix polynomial $B(z)$ of degree n that satisfies Eq. (8) and whose highest degree coefficient $A \in \mathbb{C}^{2\times 2}$ is regular. This implies that $B(z) \equiv B^\star(z)A$. We will show below that $\det A = 1$. Note that only the second row of $B^\star(z)$ is needed to compute the inversion parameters. If we compute the inversion parameters from the polynomials that appear in the second row of $B(z)$, then we still obtain the correct value of L^{-1} because $\det A = 1$. Indeed, suppose that

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

and let $p'(z) := ap(z) + cu(z)$ and $u'(z) := bp(z) + du(z)$. Define

$$\begin{bmatrix} u'_1 \\ \vdots \\ u'_n \end{bmatrix} = -\frac{1}{n} D_{n,\omega} \begin{bmatrix} u'(y_1) \\ \vdots \\ u'(y_n) \end{bmatrix}, \qquad \begin{bmatrix} p'_1 \\ \vdots \\ p'_n \end{bmatrix} = \frac{1}{n} D_{n,\omega} \begin{bmatrix} p'(y_1) \\ \vdots \\ p'(y_n) \end{bmatrix},$$

$$\begin{bmatrix} \tilde{u}'_1 \\ \vdots \\ \tilde{u}'_n \end{bmatrix} = -\frac{1}{n} \overline{\zeta^{n-1}} D_{n,\omega} \begin{bmatrix} u'(z_1) \\ \vdots \\ u'(z_n) \end{bmatrix}, \qquad \begin{bmatrix} \tilde{p}'_1 \\ \vdots \\ \tilde{p}'_n \end{bmatrix} = \frac{1}{n} \overline{\zeta^{n-1}} D_{n,\omega} \begin{bmatrix} p'(z_1) \\ \vdots \\ p'(z_n) \end{bmatrix}.$$

Then

$$u'_k = -bp_k + du_k, \qquad p'_k = ap_k - cu_k,$$
$$\tilde{u}'_k = -b\tilde{p}_k + d\tilde{u}_k, \qquad \tilde{p}'_k = a\tilde{p}_k - c\tilde{u}_k$$

for $k = 1, \ldots, n$. This implies that

$$\left[\frac{\tilde{u}_k' p_l' - \tilde{p}_k' u_l'}{y_l - z_k}\right]_{k,l=1}^{n} = \left[\frac{\tilde{u}_k p_l - \tilde{p}_k u_l}{y_l - z_k}\right]_{k,l=1}^{n}$$

because

$$\begin{aligned}\tilde{u}_k' p_l' - \tilde{p}_k' u_l' &= \det\begin{bmatrix}\tilde{u}_k' & u_l' \\ \tilde{p}_k' & p_l'\end{bmatrix} = \det\left(\begin{bmatrix} d & -b \\ -c & a\end{bmatrix}\begin{bmatrix}\tilde{u}_k & u_l \\ \tilde{p}_k & p_l\end{bmatrix}\right) \\ &= \det\begin{bmatrix}\tilde{u}_k & u_l \\ \tilde{p}_k & p_l\end{bmatrix} = \tilde{u}_k p_l - \tilde{p}_k u_l\end{aligned}$$

for $k, l = 1, \ldots, n$ since

$$\begin{bmatrix} d & -b \\ -c & a\end{bmatrix}\begin{bmatrix} a & b \\ c & d\end{bmatrix} = \begin{bmatrix}\det A & 0 \\ 0 & \det A\end{bmatrix} = I_2.$$

Algorithm 7.

input $n; y_1, \ldots, y_n, z_1, \ldots, z_n; c_1, \ldots, c_n, d_1, \ldots, d_n$

output $B(z), \alpha > 0$

initialization

$$B(z) \leftarrow \begin{bmatrix}1 & 0 \\ 0 & 1\end{bmatrix}; \quad s \leftarrow \begin{bmatrix} y_1 \\ \vdots \\ y_n \\ z_1 \\ \vdots \\ z_n\end{bmatrix}; \quad L \leftarrow \begin{bmatrix} 1 \\ \vdots \\ 1 \\ 1 \\ \vdots \\ 1\end{bmatrix}; \quad R \leftarrow \begin{bmatrix} -c_1 \\ \vdots \\ -c_n \\ -d_1 \\ \vdots \\ -d_n\end{bmatrix}$$

status $\leftarrow$ 1

comment

status = 1: both the left and the right choice are possible

status = 2: only the left choice is possible

status = 3: only the right choice is possible

flow = 1: L contains the pivot element

flow = 2: R contains the pivot element

1. $\alpha \leftarrow \max_{1 \le k \le 2n} \{\max\{|\Re R(k)|, |\Im R(k)|\}\}$
2. $R \leftarrow R/\alpha$

for $j = 1{:}2n$

select case (status)

case (1)

3.1a[maxR, pivR] $\leftarrow \max_{j \le k \le 2n} \{\max\{|\Re R(k)|, |\Im R(k)|\}\}$

3.1b[maxL, pivL] $\leftarrow \max_{j \le k \le 2n} \{\max\{|\Re L(k)|, |\Im L(k)|\}\}$

if maxR > maxL **then**

3.1c piv $\leftarrow$ pivR; flow $\leftarrow$ 2: status $\leftarrow$ 2

else
3.1d piv $\leftarrow$ pivL; flow $\leftarrow$ 1; status $\leftarrow$ 3
end if
case (2)
3.2a [maxL, pivL] $\leftarrow \max_{j \leq k \leq 2n} \{\max\{|\Re L(k)|, |\Im L(k)|\}\}$
3.2b piv $\leftarrow$ pivL; flow $\leftarrow$ 1; status $\leftarrow$ 1
case (3)
3.3a [maxR, pivR] $\leftarrow \max_{j \leq k \leq 2n} \{\max\{|\Re R(k)|, |\Im R(k)|\}\}$
3.3b piv $\leftarrow$ pivR; flow $\leftarrow$ 2; status $\leftarrow$ 1
end select
$s(j) \leftrightarrow s(\text{piv}); L(j) \leftrightarrow L(\text{piv}); R(j) \leftrightarrow R(\text{piv})$
select case (flow)
case (1)
4.1a $\mu \leftarrow R(j)/L(j)$
4.1b $B(z) \leftarrow B(z)\begin{bmatrix} z - s_j & -\mu \\ 0 & 1 \end{bmatrix}$
for $k = j + 1{:}2n$
4.1c $L(k) \leftarrow (s_k - s_j)L(k)$
4.1d $R(k) \leftarrow -\mu L(k) + R(k)$
end for
case (2)
4.2a $\mu \leftarrow L(j)/R(j)$
4.2b $B(z) \leftarrow B(z)\begin{bmatrix} 1 & 0 \\ -\mu & z - s_j \end{bmatrix}$
for $k = j + 1{:}2n$
4.2c $L(k) \leftarrow L(k) - \mu R(k)$
4.2d $R(k) \leftarrow (s_k - s_j)R(k)$
end for
end select
end for

Notes. This algorithm is a straightforward adaptation of an algorithm that appeared in [42]. Therefore we state it without proof.

We have written down steps 3.1a, 3.1b, 3.2a and 3.3a using standard Matlab notation: the maxima are computed together with their location.

The scaling in step 2 is done for reasons of numerical stability. If we divide the interpolation ordinates $c_1, \ldots, c_n, d_1, \ldots, d_n$ by α, then we do not solve the original interpolation problem, of course, but rather the one that is connected with the Loewner matrix L/α. By applying the inversion formula we now obtain the matrix αL^{-1} and thus $x' = L^{-1}b'$ is given by $x' = \tilde{x}'/\alpha$ where $\tilde{x}' := (\alpha L^{-1})b'$.

If the algorithm first performs a "left" step (flow = 1) and then a "right" step (flow = 2), then the highest degree coefficient (hdc) of $B(z)$ is multiplied on the right by

$$\operatorname{hdc}\begin{bmatrix} z-s & -\mu \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ -\mu' & z-s' \end{bmatrix} = \begin{bmatrix} 1 & -\mu \\ 0 & 1 \end{bmatrix}.$$

If, on the other hand, the algorithm first performs a right step and then a left step, then the hdc of $B(z)$ is multiplied on the right by

$$\operatorname{hdc}\begin{bmatrix} 1 & 0 \\ -\mu & z-s \end{bmatrix}\begin{bmatrix} z-s' & -\mu' \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -\mu & 1 \end{bmatrix}.$$

Together with the initialization of $B(z)$ this implies that the determinant of the hdc of $B(z)$ is equal to 1.

To calculate the inversion parameters we first have to evaluate the polynomials that appear in the second row of $B(z)$ in the points $y_1,\ldots,y_n$ and $z_1,\ldots,z_n$, which (up to a permutation) are equal to the $2n$th roots of unity $1,\zeta,\ldots,\zeta^{2n-1}$ and thus we can use the FFT.

What is the arithmetic complexity of this algorithm? As we already said, only the second row of $B(z)$ needs to be computed. This takes $2n^2+\mathrm{O}(n)$ complex multiplications and $2n^2+\mathrm{O}(n)$ complex additions. The updating of the elements of L and R, the residuals, costs $4n^2+\mathrm{O}(n)$ complex multiplications and $4n^2+\mathrm{O}(n)$ complex additions. Therefore the overall cost is $6n^2+\mathrm{O}(n)$ complex multiplications and $6n^2+\mathrm{O}(n)$ complex additions.

The multipliers μ that have to be computed in step 4.1a or 4.2a are well defined. Indeed, consider the set $\mathcal{S}$ of all the column vector polynomials $w(z)\in\mathbb{C}[z]^{2\times 1}$ that satisfy the interpolation conditions

$$f_k^{\mathrm{T}} w(s_k) = 0, \quad k = 1,\ldots,2n. \tag{9}$$

If after the execution of step $j<2n$ all the residuals in R or L would be equal to zero, then the interpolation problem (9) would have a solution of degree $<n$. This is impossible, as we will now show. The set $\mathcal{S}$ forms a submodule of the $\mathbb{C}[z]$-module $\mathbb{C}[z]^{2\times 1}$. A basis for $\mathcal{S}$ always consists of exactly two elements [43], Theorem 3.1. Let $\{B_1(z),B_2(z)\}$ be a basis for $\mathcal{S}$. Then every element $w(z)\in\mathcal{S}$ can be written in a unique way as $w(z)=\alpha_1(z)B_1(z)+\alpha_2(z)B_2(z)$ with $\alpha_1(z),\alpha_2(z)\in\mathbb{C}[z]$. The matrix polynomial $B(z):=[B_1(z)\ B_2(z)]\in\mathbb{C}[z]^{2\times 2}$ is called a *basis matrix*. Basis matrices can be characterized as follows.

Theorem 8. *A matrix polynomial* $C(z)=[C_1(z)\ C_2(z)]\in\mathbb{C}[z]^{2\times 2}$ *is a basis matrix if and only if* $C_1(z),C_2(z)\in\mathcal{S}$ *and* $\deg\det C(z)=2n$.

Proof. This follows immediately from [43], Theorem 4.1. □

Note that $B^\star(z)$ is a basis matrix.

A matrix polynomial is called *column reduced* if the highest degree coefficients of its column vector polynomials are linearly independent. Every basis matrix can be transformed into a column reduced basis matrix [43], p. 455. Note that $B^{\star}(z)$ is column reduced.

Theorem 9. *Let* $\delta_1 := \deg B_1(z)$ and $\delta_2 := \deg B_2(z)$. *If* $B(z)$ *is column reduced, then every element* $w(z) \in \mathcal{S}$ *having degree* $\leqslant \delta$ *can be written in a unique way as* $w(z) = \alpha_1(z)B_1(z) + \alpha_2(z)B_2(z)$ *with* $\alpha_1(z) \in \mathbb{C}[z]$, $\deg\alpha_1(z) \leqslant \delta - \delta_1$ *and* $\alpha_2(z) \in \mathbb{C}[z]$, $\deg\alpha_2(z) \leqslant \delta - \delta_2$.

Proof. See [43], Theorem 3.2. □

Corollary 10. *The interpolation problem* (9) *has no nontrivial solution of degree* $< n$.

Proof. $B^{\star}(z)$ is a column reduced basis matrix. Its column degrees are equal to n. Suppose that $\delta < n$. Then $\deg\alpha_1(z) < 0$ and $\deg\alpha_2(z) < 0$ and thus $\alpha_1(z) \equiv 0$ and $\alpha_2(z) \equiv 0$. This implies that $w(z) \equiv [0\ 0]^{\mathrm{T}}$. □

5. Numerical examples

We have implemented our Hankel solver in Fortran 90 and in Matlab. [2] In the Fortan version the FFTs are calculated via FFTPACK.

As a matrix-vector product involving a Hankel matrix amounts to a convolution of two vectors or, equivalently, the product of two polynomials, the residue $r := b - H\hat{x}$ can be calculated via FFT in $O(n \log n)$ flops [2]. This implies that improving an approximation for x iteratively does not add to the $O(n^2)$ complexity of our algorithm.

In the following examples we have used the Fortran 90 package. The calculations have been done by an IBM SP2 with machine epsilon $\epsilon \approx 0.12 \times 10^{-6}$ in single precision and $\epsilon \approx 0.22 \times 10^{-15}$ in double precision.

Example 11. We considered single precision $n \times n$ real Hankel matrices H_n whose entries were random uniformly distributed in $[-1, 1]$ for $n = 10(10)500$. The right-hand sides $b_n \in \mathbb{R}^n$ were calculated in double precision such that $x_n := H_n^{-1}b_n = [1 \cdots 1]^{\mathrm{T}}$. Fig. 1 shows the results obtained by our algorithm (before and after one step of iterative improvement in which the residue was calculated in double precision) and the results obtained via Gaussian elimination with partial pivoting (GEPP) using the LAPACK routines CGETRF and CGETRS. We have plotted $-\log_{10}(\|\hat{x}_n - x_n\|_\infty / \|x_n\|_\infty)$. Also

[2] The Fortran 90 package is available at `http://www.cs.kuleuven.ac.be/~marc hankel/`. The Matlab m-files can be obtained via anonymous `ftp` to `ftp.mathworks.com`. Look for the files `hsolve.m` and `ratint.m` in the directory `/pub/contrib/linalg`.

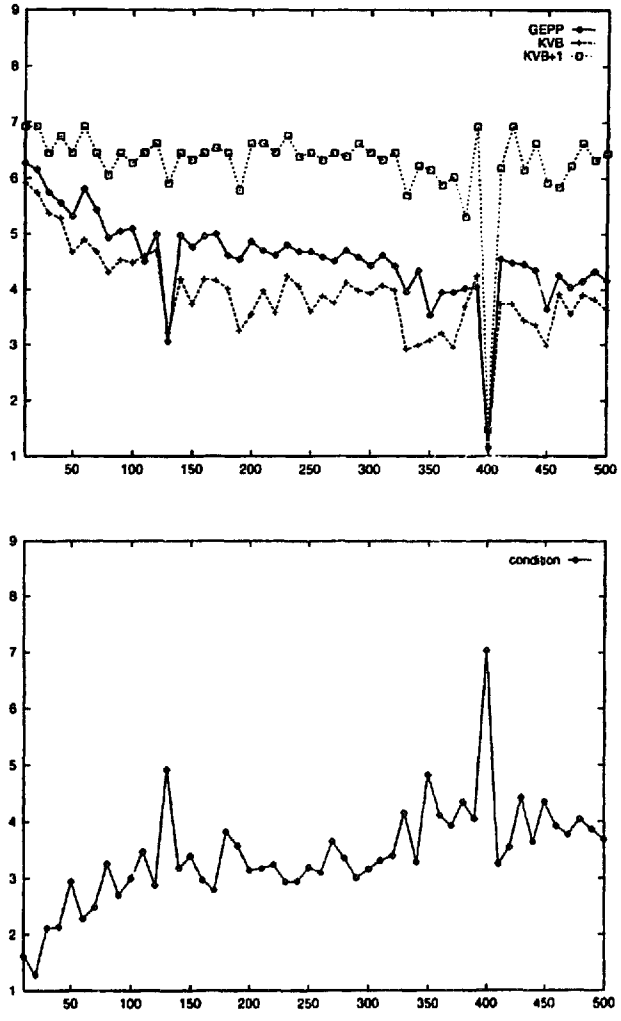

Fig. 1. $-\log_{10}\frac{\|\hat{x}_n - x_n\|_\infty}{|x_n|_\infty}$ and $\log_{10}\kappa_\infty(H_n)$ versus n.

shown is $\log_{10}\kappa_\infty(H_n)$. The condition number was calculated via LAPACK's routine CGECON. Note that after one step of iterative improvement the relative accuracy of the solution is of the order of the machine precision except for very ill-conditioned Hankel matrices.

Example 12. In this test we considered the Hankel matrix

$$H := \begin{bmatrix} (\frac{1}{2})^{n-1} & \cdots & \cdots & \frac{1}{2} & \epsilon \\ (\frac{1}{2})^{n-2} & & & \iddots & \frac{1}{2} \\ \vdots & & \iddots & & \vdots \\ \frac{1}{2} & \iddots & & & (\frac{1}{2})^{n-2} \\ \epsilon & \frac{1}{2} & \cdots & (\frac{1}{2})^{n-2} & (\frac{1}{2})^{n-1} \end{bmatrix} \in \mathbb{R}^{n\times n}.$$

The corresponding Toeplitz matrix belongs to the class of so-called Kac-Murdock-Szegö matrices (with $\rho = \frac{1}{2}$) [34]. For $n = 3k + 1$, $k = 1, 2, \ldots$, this matrix is singular if $\epsilon = 0$ and, consequently, ill-conditioned if ϵ is set to a small number. We took $n = 1000$ and let $\epsilon = 10^{-q}$ for $q = 0, 1, \ldots, 15$. We defined the right hand side $b := [b_1 \cdots b_n]^T \in \mathbb{R}^n$ as

$$b_k := 2 + \epsilon - \left(\frac{1}{2}\right)^{k-1} - \left(\frac{1}{2}\right)^{n-k}, \quad k = 1, \ldots, n,$$

because then $x := H^{-1}b$ is equal to $[1 \cdots 1]^T$ as one can easily verify. Fig. 2 shows the results obtained by our algorithm (with up to three steps of iterative improvement). The calculations were done in double precision. We have plotted $-\log_{10}(\|\hat{x} - x\|_\infty / \|x\|_\infty)$ versus $q = -\log_{10}\epsilon$. Also shown is $\log_{10}\kappa_\infty(H)$. In Fig. 3 we have plotted $-\log_{10}(\|b - H\hat{x}\|_\infty / \|b\|_\infty)$. Note that when the condi-

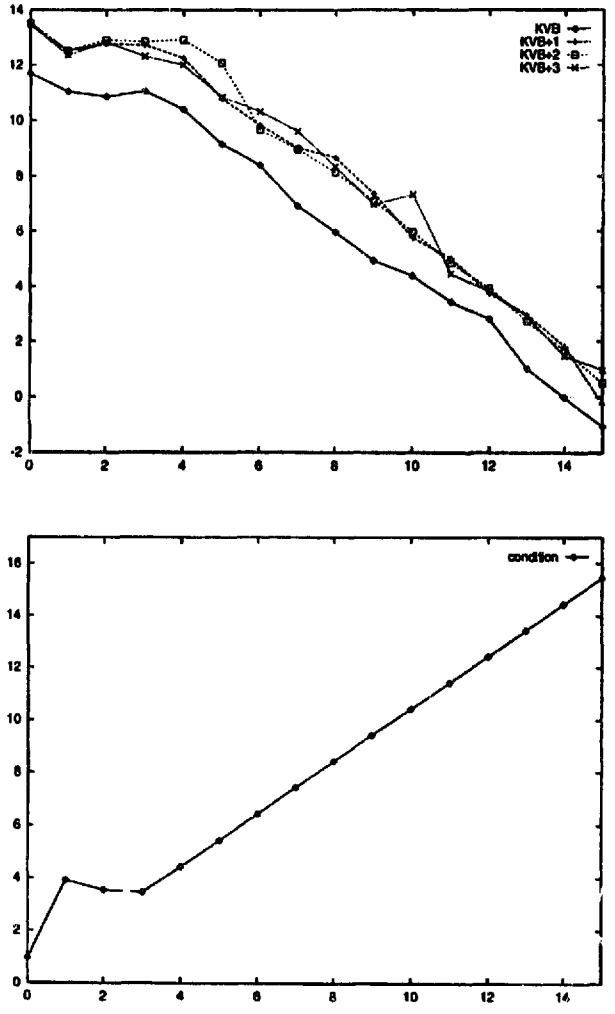

Fig. 2. $-\log_{10}\dfrac{\|\hat{x} - x\|_\infty}{|x_n|_\infty}$ and $\log_{10}\kappa_\infty(H)$ versus $-\log_{10}\epsilon$.

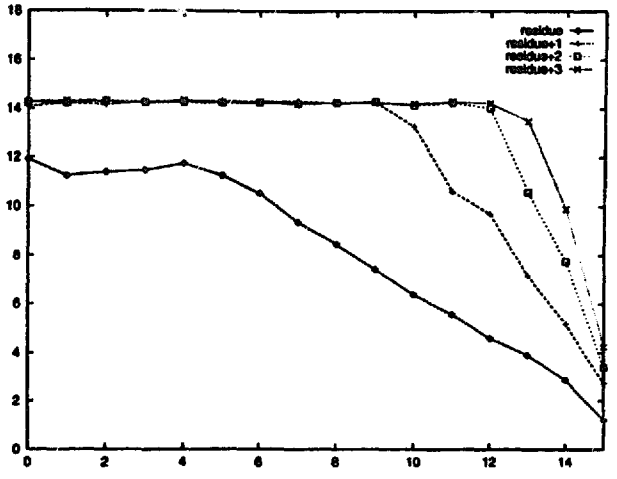

Fig. 3. $-\log_{10}\frac{\|b-H\hat{x}\|_\infty}{\|b\|_\infty}$ versus $-\log_{10}\epsilon$.

tion number is not too large, the relative error on the residue after one (or more) step(s) of iterative improvement is of the order of the machine precision.

Example 13. We considered the Hankel matrix

$$H_n := \begin{bmatrix} 1 & 2 & \cdots & n \\ 2 & & & 0 \\ \vdots & & & \vdots \\ n & 0 & \cdots & 0 \end{bmatrix}$$

for $n = 1000(100)10\,000$. One can easily verify that $\|H_n^{-1}\|_\infty = O(1/n)$. As $\|H_n\|_\infty = n(n+1)/2$, this implies that $\kappa_\infty(H_n) = O(n)$. We defined the right hand side $b_n = [\,b_1^{(n)} \ \cdots \ b_n^{(n)}\,]^T$ as

$$b_k^{(n)} := \tfrac{1}{2}n(n+1) - \tfrac{1}{2}(k-1)k, \quad k = 1, \ldots, n.$$

Then the solution to $H_n x_n = b_n$ is given by $x_n = [\,1 \cdots 1\,]^T$ as one can easily verify. Fig. 4 shows the results obtained by our algorithm (with up to two steps of iterative improvement). The calculations were done in double precision. The figures indicate that our algorithm combined with one step of iterative improvement stays weakly stable for larger sizes of the Hankel system.

6. Conclusion

In this paper we have designed a fast algorithm to solve Hankel systems. The Hankel system is transformed into a Loewner system. This requires $O(n \log n)$ flops for an $n \times n$ system. The parameters of an inversion formula for the

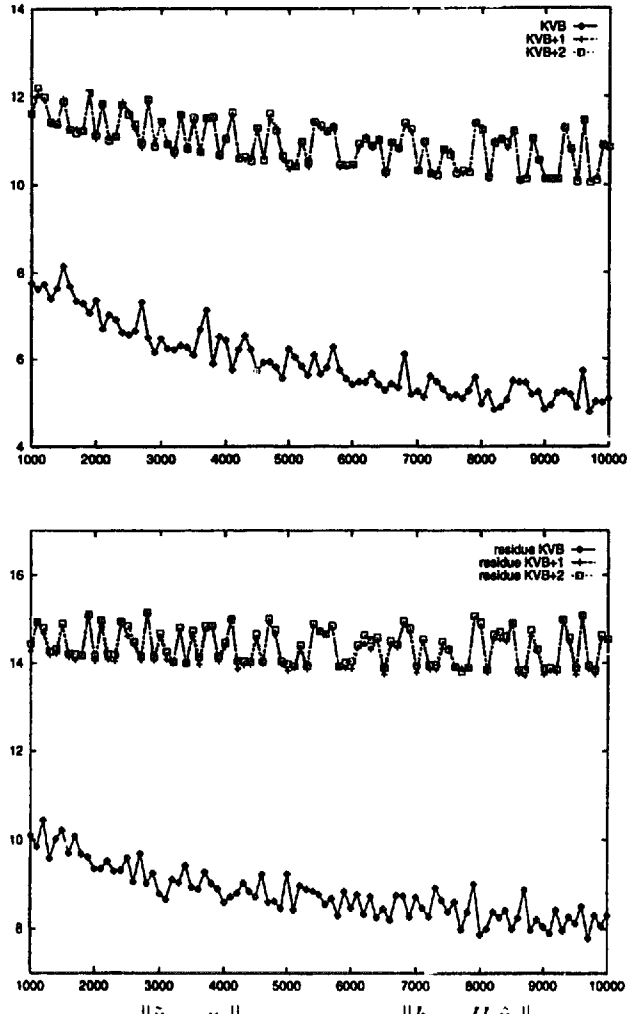

Fig. 4. $-\log_{10}\frac{\|\tilde{x}_n - x_n\|_\infty}{\|x_n\|_\infty}$ and $-\log_{10}\frac{\|b_n - H_n\tilde{x}_n\|_\infty}{\|b_n\|_\infty}$ versus n.

Loewner matrix can be computed in $O(n \log n)$ flops once the solutions to two linearized rational interpolation problems are known. Solving these problems requires $6n^2 + O(n)$ complex multiplications and $6n^2 + O(n)$ complex additions. Each step of iterative improvement can be executed very cheaply requiring only $O(n \log n)$ flops.

The numerical experiments indicate that our approach (combined with one (or more) step(s) of iterative improvement) could be weakly stable. The operation count and the accuracy obtained in the experiments show that it is competitive with existing methods.

All steps of the solution procedure can be generalized to block Hankel matrices [38].

Acknowledgements

The authors thank Georg Heinig for providing them with Refs. [11,25,28,30,29,27]. They also want to thank the referees for their helpful suggestions and comments.

References

[1] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, D. Sorensen, LAPACK users' guide, SIAM, 1994.

[2] D. Bini, V. Pan, Polynomial and Matrix Computations, vol. 1: Fundamental Algorithms, Birkhäuser, 1994.

[3] A.W. Bojanczyk, G. Heinig, A multi-step algorithm for Hankel matrices, J. Complexity 10 (1) (1994) 142–164.

[4] T. Boros, T. Kailath, V. Olshevsky, Predictive Pivoting and Backward Stability of Fast Cauchy Solvers, Unpublished manuscript, 1995.

[5] S. Cabay, A. Jones, G. Labahn, Computation of numerical Padé-Hermite and simultaneous Padé systems: I Near inversion of generalized Sylvester matrices, SIAM J. Matrix Anal. Appl. 17 (1996) 248–267.

[6] S. Cabay, A. Jones, G. Labahn, Computation of numerical Padé–Hermite and simultaneous Padé systems: II A weakly stable algorithm, SIAM J. Matrix Anal. Appl. 17 (1996) 268–297.

[7] S. Cabay, R. Meleshko, A weakly stable algorithm for Padé approximants and the inversion of Hankel matrices, SIAM J. Matrix Anal. Appl. 14 (3) (1993) 735–765.

[8] T.F. Chan, P.C. Hansen, A look-ahead Levinson algorithm for general Toeplitz systems, IEEE Trans. Signal Process. 40 (5) (1992) 1079–1090.

[9] T.F. Chan, P.C. Hansen, A look-ahead Levinson algorithm for indefinite Toeplitz systems, SIAM J. Matrix Anal. Appl. 13 (2) (1992) 490–506.

[10] M. Fiedler, Hankel and Loewner matrices, Linear Algebr. Appl. 58 (1984) 75–95.

[11] T. Finck, G. Heinig, K. Rost, An inversion formula and fast algorithms for Cauchy-Vandermonde matrices, Linear Algebr. Appl. 183 (1993) 179–191.

[12] R.W. Freund, A look-ahead Bareiss algorithm for general Toeplitz matrices, Numer. Math. 68 (1994) 35–69.

[13] R.W. Freund, H. Zha, Formally biorthogonal polynomials and a look-ahead Levinson algorithm for general Toeplitz systems, Linear Algebr. Appl. 188/189 (1993) 255–303.

[14] R.W. Freund, H. Zha, A look-ahead algorithm for the solution of general Hankel systems, Numer. Math. 64 (1993) 295–321.

[15] K.A. Gallivan, S. Thirumalai, P. Van Dooren, A look-ahead Schur algorithm, Proceedings of the Fifth SIAM Conference on Applied Linear Algebra (Snowbird, Utah), June 1994, pp. 450–454.

[16] K.A. Gallivan, S. Thirumalai, P. Van Dooren, V. Vermaut, High performance algorithms for Toeplitz and block Toeplitz matrices, Linear Algebr. Appl. 241–243 (1996) 343–388.

[17] I. Gohberg, T. Kailath, I. Koltracht, Efficient solution of linear systems of equations with recursive structure, Linear Algebr. Appl. 80 (1986) 81–113.

[18] I. Gohberg, T. Kailath, I. Koltracht, P. Lancaster, Linear complexity parallel algorithms for linear systems of equations with recursive structure, Linear Algebr. Appl. 88/89 (1987) 271–315.

[19] I. Gohberg, T. Kailath, V. Olshevsky, Fast Gaussian elimination with partial pivoting for matrices with displacement structure, Math. Comput. 64 (212) (1995) 1557–1576.
[20] I. Gohberg, V. Olshevsky, Fast state space algorithms for matrix Nehari and Nehari-Takagi interpolation problems, Integral Equations Operator Theory 20 (1994) 44–83.
[21] G. Golub, V. Olshevsky, Pivoting on structured matrices with applications, Unpublished manuscript, 1997.
[22] M.H. Gutknecht, Stable row recurrences for the Padé table and a generically superfast look-ahead solver for non-Hermitian Toeplitz systems, Linear Algebr. Appl. 188/189 (1993) 351–421.
[23] M.H. Gutknecht, M. Hochbruck, Look-ahead Levinson- and Schur-type recurrences in the Padé table, Electron. Trans. Numer. Anal. 2 (1994) 104–129.
[24] M.H. Gutknecht, M. Hochbruck, Look-ahead Levinson and Schur algorithms for non-Hermitian Toeplitz systems, Numer. Math. 70 (1995) 181–228.
[25] G. Heinig, Inversion of generalized Cauchy matrices and other classes of structured matrices, Linear Algebra in Signal Processing, IMA volumes in Mathematics and its Applications 69 (1994) 95–114.
[26] G. Heinig, Inversion of Toeplitz-like matrices via generalized Cauchy matrices and rational interpolation, Systems and Networks: Mathematical Theory and Applications. Vol. II: Invited and Contributed Papers, Mathematical Research, vol. 79, Akademie Verlag, Berlin, 1994, pp. 707–711.
[27] G. Heinig, Solving Toeplitz systems via extension and interpolation, CALCOLO 33 (1996) 115–129, Proceedings of the Workshop Toeplitz Matrices: Structure, Algorithms and Applications, Cortona (Italy), September 9–12, 1996.
[28] G. Heinig, Transformation approaches for fast and stable solution of Toeplitz systems and polynomial equations, Proceedings of the International Workshop, Recent Advances in Applied Mathematics (State of Kuwait), May 4–7 1996, pp. 223–238.
[29] G. Heinig, A. Bojanczyk, Transformation techniques for Toeplitz and Toeplitz-plus-Hankel matrices: II Algorithms, Linear Algebr. Appl. (submitted).
[30] G. Heinig, A. Bojanczyk, Transformation techniques for Toeplitz and Toeplitz-plus-Hankel matrices: I Transformations, Linear Algebr. Appl. 254 (1997) 193–226.
[31] G. Heinig, K. Rost, Algebraic Methods for Toeplitz-like Matrices and Operators, Operator Theory: Advances and Applications, vol. 13, Birkhäuser, 1984.
[32] N.J. Higham, Stability analysis of algorithms for solving confluent Vandermonde-like systems, SIAM J. Matrix Anal. Appl. 11 (1) (1990) 23–41.
[33] T. Huckle, A look-ahead algorithm for solving nonsymmetric linear Toeplitz equations, Proceedings of the Fifth SIAM Conference on Applied Linear Algebra, Snowbird, Utah, June 1994, pp. 455–459.
[34] M. Kac, W.L. Murdock, G. Szegö, On the eigen-values of certain Hermitian forms, J. Rational Mech. Anal. 2 (1953) 767–800.
[35] T. Kailath, V. Olshevsky, Diagonal pivoting for partially reconstructible Cauchy-like matrices, with applications to Toeplitz-like linear equations and to boundary rational matrix interpolation problems, Linear Algebr. Appl. 254 (1997) 251–302.
[36] T. Kailath, A.H. Sayed, Displacement structure: theory and applications, SIAM Review 37 (1995) 297–386.
[37] G. Kowalewski, Interpolation und Genäherte Quadratur, Teubner, 1932.
[38] P. Kravanja, M. Van Barel, A fast block Hankel solver based on an inversion formula for block Loewner matrices, CALCOLO 33 (1996) 147–164, Proceedings of the Workshop Toeplitz Matrices: Structure, Algorithms and Applications, Cortona (Italy), September 9–12, 1996.
[39] C. Van Loan, Computational frameworks for the fast Fourier transform, Frontiers in Applied Mathematics, vol. 10, SIAM, Philadelphia, 1992.

[40] K. Löewner, Über monotone Matrixfunktionen, Math. Z. 38 (1934) 177–216.
[41] L. Reichel, Newton interpolation at Leja points, BIT 30 (1990) 332–346.
[42] M. Van Barel, A. Bultheel, A new approach to the rational interpolation problem, J. Comput. Appl. Math. 32 (1990) 281–289.
[43] M. Van Barel, A. Bultheel, A general module theoretic framework for vector M-Padé and matrix rational interpolation, Numer. Algorithms 3 (1992) 451–462.
[44] M. Van Barel, A. Bultheel, A lookahead algorithm for the solution of block Toeplitz systems, Linear Algebr. Appl. 266 (1997) 291–335.
[45] M. Van Barel, Z. Vavřín, Inversion of a block Löwner matrix, J. Comput. Appl. Math. 69 (1996) 261–284.
[46] Z. Vavřín, Remarks of complexity of polynomial and special matrix computations, Linear Algebr. Appl. 122/123/124 (1989) 539–564.
[47] Z. Vavřín, Inverses of Löwner matrices, Linear Algebr. Appl. 63 (1984) 227–236.