A

On some numerical and algebraic computations with matrices and polynomials

by

ZHAO QIN CHEN

A dissertation submitted to the Graduate Faculty in Mathematics in partial fulfillment of requirements for the degree of Doctor of Philosophy, The City University of New York

2000

UMI Number: 9959171

Bell & Howell Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

This manuscript has been read and accepted for the Graduate Faculty in Mathematics in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

11/15/99 — Victor Pan
Date — Chair of Examining Committee

11/17/99
Date — Executive Officer

MICHAEL ANSHEL Michael Anshel

ALEXEI MYASNIKOV Alexei Miasnikov

VICTOR Y. PAN Victor Pan

Supervisory Committee

THE CITY UNIVERSITY OF NEW YORK

Abstract

On some numerical and algebraic computations with matrices and polynomials

by

ZHAO QIN CHEN

Advisor: Professor Victor Y. Pan

Polynomial and matrix computations constitute the bulk of the area of modern computations for sciences, engineering and communication. From mathematical point of view, these are usually algebraic problems but the effectiveness of practical algorithms for their solution can frequently be increased by involving approximation techniques and other methods of numerical computation. We demonstrate this by the examples of some important computations from three areas.

In the first chapter of the dissertation, we follow [PACLS98] and [PACPS98]. We consider Trummer's classical computational problem, having several important applications to celestial mechanics (n-body problem), fluid mechanics, computation of Riemann zeta function, conformal maps, solution of integral equations, polynomial and rational interpolation and evaluation on a fixed node set. The known fast algebraic algorithms for the solution of Trummer's problem have substantial practical deficiency – they are numerically unstable. We proceed by first reducing the solution to computations with structured matrices, then we transform these matrices into matrices with other forms of structure. With such transformations, the problem was brought into a form convenient for its fast and numerically stable approximate solutions. The links established among the cited computations enable us to perform all of them more efficiently. The resulting

algorithms remain numerically stable and fast for a larger class of input matrices versus the knowns results.

In the second chapter of the dissertation, we follow [PC99]. We consider most effective algebraic and numerical algorithms for polynomial division, which we relate to the computations with structured matrices and polynomials. A customary approach to fast polynomial division relies on FFT, Fast Fourier Transform. A certain disadvantage of this method is the involvement of computations with complex numbers even where the inputs are real. We show a simple alternative algorithm using only Discrete Sine Transform, DST (instead of FFT) based on the results of T. Kailath and V. Olshevsky presented in [KO96]. The resulting algorithms compute the output (quotient and the remainder) approximately, operate only with real numbers where the input is real, and remain as fast as the known algorithms, which achieve record complexity bound.

In the third chapter of the dissertation, we follow [PC99a]. We consider the matrix eigenproblem, which is a central problem of applied linear algebra. In the unsymmetric case, the known customary algorithms have substantial deficiency, that is, they are lack of guaranteed fast convergence and, indeed, diverge or stumble in practice in the important cases of multiple and clustered eigenvalues. (The latter cases routinely arise when the input matrix with multiple eigenvalues is perturbed by small errors.) To yield solution algorithms that remain fast even on the worst case input, we reduce the computation of the eigenvalues of a matrix to the approximatation of the roots (zeros) of its minimum or characteristic polynomials, for which one may apply the recent highly effective polynomial rootfinders (e.g. ones that yield nearly linear solution cost bound [P95]). Keller-Gehrig in [K-G85] reduced any input matrix to a similar matrix in the triangular Frobenius form, also called semicyclic Frobenius form. For generic input, the reduction is achieved even to a simpler form, the output being a Frobenius matrix. The arithmetic computational complexity of such a reduction is $O(M(n)\log n)$ for the worst case input and $O(M(n))$ for generic input provided that a pair of $n \times n$ matrices can be multiplied at the cost $O(M(n))$ (see [K-G85]). Theoretically $M(n) \leq Cn^{2.376}$ for an immense constant C, but the practical algorithms usually support $M(n) = 2n^3 - n^2$. For the generic structured and generic sparse matrices A, however, the cost bound of such a reduction decreases to $O(nM_{A\mathbf{v}}(n))$ provided that $M_{A\mathbf{v}}(n)$ denotes the cost of multiplication of the matrix A by a vector, and $nM_{A\mathbf{v}}(n)$ is of order n^2 (up to logarithmic or polylogarithmic factors) for many important classes of sparse and structured matrices. We also show a quadratic time algorithm for the subsequent computation of the bases for all the spaces of the eigenvectors associated with such matrices, which is optimal (up to a constant factor). For the general matrix case, the eigenspace computation techniques yield cubic time. We also show an alternative to the stage of the reduction to triangular Frobenius (semicyclic) form, which leads to the same overall computational cost bound. In this case, the reduction is to the tridiagonal form and is obtained by means of an unsymmetric modification of Lanczos algorithm, elaborated in [Par92] (cf. also [BP94]). The entire construction combines rational algebraic computations at the stages of the reduction of the input matrix to special simple forms and of the eigenspace computations with the numerical methods for the approximation of the eigenvalues as the roots of the characteristic (or minimal) polynomial of an input matrix.

Summarizing, in all three parts we demonstrated that combined applications of numerical and algebraic techniques is an effective tool for reducing computational cost for the solution of various algebraic tasks.

Acknowledgements

I would like to express my deepest gratitude and appreciation to my advisor Professor Victor Pan for his consistent source of encouragement and support of all kinds. He was also generous with his suggestions and comments. I also admire his constant enthusiasm, integrity and creativity.

Secondly, it was a pleasure and honor to have Professors Michael Anshel and Alexei Myasnikov serving in the Committee. I have appreciated their helpful comments and advice.

Moreover, I wish to thank Professors J. Dodziuk, R. Feinerman, M. Fitting, M. Nathanson, B. Randal, A. Vasquez and all other faculty members from the Mathematics Department of Graduate School of CUNY.

Finally, the most of all, I owe tremendous debt of gratitudes to all my family, my parents, my wife, my son, my brothers and sisters for their unconditional love, devotion, understanding and support.

Contents

Chapter 1

New Transformations of Cauchy Matrices and Trummer's Problem

1.1 Summary

We show some new expressions for a Cauchy matrix, which enable us to simplify the solution of Trummer's problem, both in the general case and in the case where the input Cauchy matrix is fixed for the problem whereas the input vector varies.

1.2 Introduction.

The solution of Trummer's problem (that is, the problem of multiplication of an $n \times n$ Cauchy matrix C by a vector) is the basis for the solution of several important problems of scientific and engineering computing [A63], [A85], [A86], [B88], [C73], [D83], [HE81], [ODR89], [OS88], [R85]. The straightforward algorithm solves Trummer's problem in $O(n^2)$ flops. The fast algorithm of [Ger87] uses $O(n \log^2 n)$ flops but has poor numerical stability. Presently the algorithm of choice in practical computations is the celebrated Multipole Algorithm [R85], [GR87], [AGR88], [CGR88], [G88], [BP94], pp. 261–262, which belongs to the class of hierarchical methods [A85], [BH86], [VDR89]. The algorithm approximates the solution in $O(n \log n)$ flops in terms of n, but its cost estimate and even its ability to yield the desired approximation at all also depend on the bound on the approximation error and on the correlation between the entries of the pair of

n-dimensional vectors defining the input matrix C.

The goal of the present chapter is to enhance the power of the Multipole Algorithm (as well as other solution algorithms for Trummer's problem) by showing some new expressions for a Cauchy matrix via other Cauchy matrices, which we may vary by changing one of their basis vectors. Under an appropriate choice of such a vector, the subsequent solution of Trummer's problem is simplified; in particular, the power of the Multipole Algorithm can be enhanced.

Technically, we achieve our goal by means of a simple transformation of the useful basic formulae of [FHR93], and our resulting expressions for C give us further algorithmic opportunities. The underlying idea of the transformation of the basic vectors defining the problem is taken from [PLST93], where this idea was used for multipoint polynomial evaluation and interpolation.

We use the following order of presentation. In the next section, we introduce the definitions, show how to avoid degeneration of Trummer's problem, and recall some basic formulae from [FHR93]. In section 1.4, we extend these formulae to yield the desired transformations of Cauchy matrices and Trummer's problem. In section 1.5 we comment on the algorithmic aspects.

1.3 Definitions, basic expressions, and treatment of degeneration.

Definition 1.3.1 *For a pair of n-dimensional vectors $\vec{a} = (a_i)_{i=0}^{n-1}$, $\vec{b} = (b_j)_{j=0}^{n-1}$, let $C(\vec{a}, \vec{b}) = (\frac{1}{a_i - b_j})_{i,j=0}^{n-1}$, $V(\vec{a}) = (a_i^j)_{i,j=0}^{n-1}$, $H(\vec{a}) = (h_{i,j})_{i,j=0}^{n-1}$, $h_{i,j} = a_{i+j}$ for $i + j \le n - 1$, $h_{i,j} = 0$ for $i + j \ge n - 1$, denote the associated $n \times n$ Cauchy, Vandermonde and triangular Hankel matrices, respectively. For a vector $\vec{a} = (a_i)_{i=0}^{n-1}$ with $a_i \ne a_j$ for $i \ne j$, a Cauchy degenerate matrix $C(\vec{a})$ has the diagonal entries zeros and the (i, j)-th entry $\frac{1}{a_i - a_j}$ for $i \ne j$. W^{-1}, W^T and W^{-T} denote the inverse, the transpose and the transpose of the inverse of a matrix W, respectively. Furthermore, $p_{\vec{b}}(x)$ and $p'_{\vec{b}}(x)$ denote the polynomial $p_{\vec{b}}(x) = \prod_{j=0}^{n-1}(x - b_j)$ and its derivative $p'_{\vec{b}}(x) = \sum_{i=0}^{n-1}\prod_{j=0(j\ne i)}^{n-1}(x - b_j)$, respectively. Finally, $D(\vec{a}, \vec{b}) = \mathrm{diag}(p_{\vec{b}}(a_i))_{i=0}^{n-1} = \mathrm{diag}(\prod_{j=0}^{n-1}(a_i - b_j))_{i=0}^{n-1}$ and $D'(\vec{b}) = \mathrm{diag}(p'_{\vec{b}}(b_j))_{j=0}^{n-1} = \mathrm{diag}(\prod_{j=0(j\ne i)}^{n-1}(b_i - b_j))_{i=0}^{n-1}$ denote a pair of $n \times n$ diagonal matrices,* *defined by the vectors $\vec{a}$ and $\vec{b}$.*

Theorem 1.3.1 *Let $c_i \neq d_j$, $i, j = 0, 1, \ldots, n-1$. Then*

$$C(\vec{c}, \vec{d}) = D(\vec{c}, \vec{d})^{-1} V(\vec{c}) H(\vec{d}) V(\vec{d})^T, \tag{3.1}$$

$$C(\vec{c}, \vec{d}) = D(\vec{c}, \vec{d})^{-1} V(\vec{c}) V(\vec{d})^{-1} D'(\vec{d}). \tag{3.2}$$

Remark 1.3.1 *Equation (3.1) is taken from [FHR93]. Equation (3.2) is implicit in [Ger87].*

Definition 1.3.2 *Trummer's problem is the problem of computing the vector $C(\vec{a}, \vec{b})\vec{v}$ for three given vectors $\vec{a} = (a_i)_{i=0}^{n-1}$, $\vec{b} = (b_j)_{j=0}^{n-1}$ and $\vec{v} = (v_j)_{j=0}^{n-1}$, where $a_i \neq b_j$ for all pairs i, j. Trummer's degenerate problem is the problem of computing the vector $C(\vec{a})\vec{v}$ for two given vectors $\vec{a} = (a_i)_{i=0}^{n-1}$ and $\vec{v} = (v_j)_{j=0}^{n-1}$, where $a_i \neq a_j$ for $i \neq j$.*

Definition 1.3.3 *$\omega_k = \exp(2\pi\sqrt{-1}/k)$ is a primitive k-th root of 1, $\omega_k^k = 1, \omega_k^l \neq 1$ for $l = 1, \ldots, k-1$.*

Lemma 1.3.1 *$\sum_{l=0}^{k-1} \omega_k^{gl} = 0$ for $g = 1, \ldots, k-1$.*

Approximate solution of Trummer's degenerate problem can be reduced to Trummer's problem due to the next simple result.

Lemma 1.3.2 *$C(\vec{c}) = \frac{1}{h}\sum_{g=0}^{h-1} C(\vec{c}, \vec{c} + \epsilon\omega_h^g \vec{e}) + O(\epsilon^h)$ as $\epsilon \to 0$, where $\vec{e} = (1)_{j=0}^{n-1}$ is the vector filled with the values one and where ϵ is a scalar parameter.*

Proof. $\sum_{g=0}^{h-1} \frac{1}{c_i - c_j - \epsilon\omega_h^g} = \frac{1}{c_i - c_j} \sum_{l=0}^{\infty} \sum_{g=0}^{h-1} (\frac{\epsilon\omega_h^g}{c_i - c_j})^l = \frac{h}{c_i - c_j}(1 + O(\epsilon^h))$, the lemma is immediately followed from lemma 1.3.1. □

1.4 New transformations of a Cauchy matrix and of Trummer's problem.

Theorem 1.4.1 *For a triple of n-dimensional vectors $\vec{b} = (b_i)_{i=0}^{n-1}$, $\vec{c} = (c_j)_{j=0}^{n-1}$, $\vec{d} = (d_k)_{k=0}^{n-1}$, where $b_i \neq c_j$, $c_j \neq d_k$, $d_k \neq b_i$ for $i, j, k = 0, \ldots, n-1$, we have the following matrix equations:*

$$C(\vec{c}, \vec{d}) = D(\vec{c}, \vec{d})^{-1} V(\vec{c}) V(\vec{b})^{-1} D(\vec{b}, \vec{d}) C(\vec{b}, \vec{d}), \tag{4.1}$$

$$C(\vec{c},\vec{d}) = D(\vec{c},\vec{d})^{-1}D(\vec{c},\vec{b})C(\vec{c},\vec{b})D'(\vec{b})^{-1}D(\vec{b},\vec{d})C(\vec{b},\vec{d}), \tag{4.2}$$

$$C(\vec{c},\vec{d}) = C(\vec{c},\vec{b})D(\vec{b},\vec{c})V(\vec{b})^{-T}V(\vec{d})^{T}D(\vec{d},\vec{c})^{-1}, \tag{4.3}$$

$$C(\vec{c},\vec{d}) = -C(\vec{c},\vec{b})D(\vec{b},\vec{c})D'(\vec{b})^{-1}C(\vec{b},\vec{d})D(\vec{d},\vec{b})D(\vec{d},\vec{c})^{-1}. \tag{4.4}$$

Proof. From (3.1), we immediately deduce that $C(\vec{b},\vec{d})^{-1} = V(\vec{d})^{-T}H(\vec{d})^{-1}V(\vec{b})^{-1}$ $D(\vec{b},\vec{d})$. Substitute the latter matrix equation and the expression (3.1) for $C(\vec{c},\vec{d})$ into the trivial matrix identity $C(\vec{c},\vec{d}) = C(\vec{c},\vec{d})C(\vec{b},\vec{d})^{-1}C(\vec{b},\vec{d})$ and obtain (4.1). Extend (3.2) to a similar expression $C(\vec{c},\vec{b}) = D(\vec{c},\vec{b})^{-1}V(\vec{c})V(\vec{b})^{-1}D'(\vec{b})$ and deduce that $V(\vec{c})V(\vec{b})^{-1} = D(\vec{c},\vec{b})C(\vec{c},\vec{b})D'(\vec{b})^{-1}$. Substitute this expression into (4.1) and obtain (4.2). Observe that $C(\vec{c},\vec{d}) = -C(\vec{d},\vec{c})^{T}$ and extend (4.1) to obtain that $-C(\vec{c},\vec{d}) = C(\vec{d},\vec{c})^{T} = (D(\vec{d},\vec{c})^{-1}V(\vec{d})$
$V(\vec{b})^{-1}D(\vec{b},\vec{c})C(\vec{b},\vec{c}))^{T} = C(\vec{b},\vec{c})^{T}D(\vec{b},\vec{c})V(\vec{b})^{-T}V(\vec{d})^{T}D(\vec{d},\vec{c})^{-1}$. Substitute $C(\vec{c},\vec{b}) = -C(\vec{b},\vec{c})^{T}$ and obtain (4.3). Finally, extend (3.2) to obtain that $V(\vec{d})V(\vec{b})^{-1} = D(\vec{d},\vec{b})$ $C(\vec{d},\vec{b})D'(\vec{b})^{-1}$ and consequently $V(\vec{b})^{-T}V(\vec{d})^{T} = D'(\vec{b})^{-1}C(\vec{d},\vec{b})^{T}\ D(\vec{d},\vec{b})$. Substitute the latter matrix equation and the matrix equation $C(\vec{d},\vec{b})^{T} = -C(\vec{b},\vec{d})$ into (4.3) and obtain (4.4). □

1.5 Some algorithmic aspects.

The expressions (4.2) and (4.4) for $C(\vec{c},\vec{d})$ are Vandermonde-free and Hankel-free, but they enable us to transform the basis vectors $\vec{c}$ and $\vec{d}$ for the Cauchy matrix $C(\vec{c},\vec{d})$ into the two pairs of basis vectors $\vec{c}$, $\vec{b}$ and $\vec{b}$, $\vec{d}$ for any choice of the vector $\vec{b} = (b_j)$, $b_j \neq c_j$, $b_j \neq d_k$, $i, j, k = 0, \ldots, n-1$. The associated Trummer's problem is reduced to

a) the evaluation of the diagonal entries of the diagonal matrices $D'(\vec{b})^{-1}$, $D(\vec{f},\vec{g})$ and/or $D(\vec{f},\vec{g})^{-1}$, for $(\vec{f},\vec{g})$ denoting the pairs $(\vec{c},\vec{d})$, $(\vec{b},\vec{d})$, $(\vec{c},\vec{b})$, $(\vec{b},\vec{c})$, $(\vec{d},\vec{b})$ and/or $(\vec{d},\vec{c})$,

b) recursive multiplication of these matrices and the Cauchy matrices $C(\vec{b},\vec{d})$ and $C(\vec{c},\vec{b})$ by vectors.

Let us next specify parts a) and b).

a) The evaluation of the entries of the matrices $D(\vec{f},\vec{g})$ and $D(\vec{f},\vec{g})^{-1}$ for a given pair of vectors $(\vec{f},\vec{g})$ and of the matrix $D'(\vec{g})$ for a given vector $\vec{g}$ can be reduced to the computation of the coefficients of the polynomial $p_{\vec{g}}(x) = \prod_{j=0}^{n-1}(x - g_j)$ and the subsequent evaluation of $p_{\vec{g}}(x)$ at the points f_i, $i = 0, \dots, n-1$ (for $D(\vec{f},\vec{g})$) and of its derivative $p'_{\vec{g}}(x)$ at the points g_i, $i = 0, \dots, n-1$ (for $D'(\vec{g})$).

   The coefficients of the polynomial $p_{\vec{g}}(x)$ can be obtained by the fan-in method, consisting of the pairwise multiplication of the linear factors $x - g_j$ followed by recursive pairwise multiplication of the computed products (cf. [BP94], p. 25). The computation is numerically stable and uses $O(n \log^2 n)$ flops.

   Multipoint polynomial evaluation can be also done in $O(n \log^2 n)$ flops ([BP94], p. 26), but due to the potential numerical stability problems, it seems more attractive to apply the more recent techniques of *fast multipoint polynomial approximation* [R88], [P95], [PLST93], [PZHY97]. We may very much simplify the evaluation of the matrices $D(\vec{f},\vec{g})$, $D(\vec{f},\vec{g})^{-1}$ and $D'(\vec{b})$, where $\vec{f} = \vec{b}$ or $\vec{g} = \vec{b}$ provided that we may choose a vector $\vec{b} = (b_i)_{i=0}^{n-1}$ at our convenience. For instance, let us fill this vector with the scaled n-th roots of 1, so that

$$b_i = a\omega_n^i, \quad i = 0, 1, \dots, n-1, \tag{5.1}$$

   for a scalar a and for ω_n of definition 1.3.3. Then $p_{\vec{b}}(x) = \prod_{i=0}^{n-1}(x - a\omega_n^i) = x^n - a^n$, $p'_{\vec{b}}(x) = nx^{n-1}$, and the matrices $D(\vec{f},\vec{b})$ and $D(\vec{b})$ can be immediately evaluated in $O(n \log n)$ flops. Furthermore, the evaluation of any given polynomial $p(x)$ of degree n at the scaled n-th roots of 1 is immediately reduced to discrete Fourier transform and thus can be performed in $O(n \log n)$ flops by means of FFT.

   Finally, all the diagonal matrices involved in (4.1)–(4.4) can be precomputed once and for all if, in Trummer's problem of the computation of the vector $C(\vec{c},\vec{d})\vec{v}$, the Cauchy matrix $C(\vec{c},\vec{d})$ is fixed (e.g. $C(\vec{v},\vec{d})$ is the Hilbert matrix $(\frac{1}{i+j+1})_{i,j=0}^{n-1}$), and only the vector $\vec{v}$ varies.

b) The multiplication of the diagonal matrices by vectors is a trivial task. The multiplication of the Cauchy matrix $C(\vec{b},\vec{d})$ or $C(\vec{c},\vec{b})$ by a vector is Trummer's problem, whose solution can be simplified under an appropriate choice of the vector $\vec{b}$. In particular, even if we restrict $\vec{b}$ to be filled with scaled roots of 1 (cf. (5.1)), we still may choose the scaling parameter a to guarantee fast convergence of the power series of the Multipole Algorithm.

The above study can be extended to the expressions (4.1) and (4.3) for $C(\vec{c}, \vec{d})$. Each of them involves two Vandermonde matrices, but one of these matrices in each expression is defined by a vector $\vec{b}$ of our choice, and this enables us to yield simplification. In particular, for two given vectors $\vec{u} = (u_i)_{i=0}^{n-1}$ and $\vec{b} = (b_i)_{i=0}^{n-1}$ the vector $\vec{v} = V(\vec{b})^{-1}\vec{u}$ is the coefficient vector of the polynomial $v(x)$ that takes on the values u_k at the points b_k, $k = 0, \ldots, n-1$. For b_k being a scaled n-th roots of 1, as in (5.1), the computation of $\vec{v}$ takes $O(n \log n)$ ops due to the inverse FFT. Similar comments apply to the multiplication of the matrix $V(\vec{b})^{-T}$ by a vector.

Remark 1.5.1 *Lemma 1.3.2 enables us to extend the above analysis to approximate solution of Trummer's degenerate problem.*

Remark 1.5.2 *By Tellegen's theorem [PSD70], the exact multiplication of the transposed Vandermonde matrix $V^T(\vec{d})$ by a vector (cf. (3.1), (4.3)) can be reduced to the exact multiplication of $V(\vec{d})$ by a vector, that is, to exact multipoint polynomial evaluation, though Tellegen's theorem does not generally preserve the error bounds of algorithms for multipoint polynomial approximation, such as ones of [R88], [P95], [PLST93], [PZHY97].*

Chapter 2

Approximate Real Polynomial Division via Approximate Inversion of Real Triangular Toeplitz Matrices

2.1 Summary.

We first modify the known FFT based algorithms for approximate polynomial division (which is equivalent to inversion of triangular Toeplitz) matrices by replacing FFT by DST, Discrete Sine Transform. The algorithm remains as fast as before but avoids involving nonreal numbers where the input is real. Then in similar modifications we replace FFT by DCT, Discrete Cosine Transform.

2.2 Introduction.

In this chapter, we will use polynomial division (which is equivalent to inversion of a triangular Toeplitz matrix) to demonstrate the correlation between the effective known algebraic and numerical algorithms with structured matrices. The $K \times K$ downshift matrix $Z = (z_{i,j})$, $z_{i+1,i} = 1$, $z_{i,j} = 0$ if $i + 1 \neq j$, $Z^K = 0$, generates the algebra of lower triangular Toeplitz matrices, $T = \sum_{i=0}^{K-1} t_i Z^i$, which is isomorphic to the ring of polynomials modulo z^K, $t(z) = \sum_{i=0}^{K-1} t_i z^i$. In particiular, $T^{-1} = \sum_{i=0}^{K-1} t_i^- Z^i$ iff $\sum_{i=0}^{K-1} t_i^- z^i t(z) = 1 \bmod z^K$, that is, the inversion of T is equivalent to computing the reciprocal of $t(z)$ modulo z^n, the latter operation is immedialtely extended to polynomial division [BP86]. To approximate T^{-1}, one may apply FFT to diagonalize and then invert an ϵ-circulant matrix C_ϵ approximating T [B84]. Such an FFT based algorithm uses only $O(K \log K)$ flops and allows its efficient parallelization but involves nonreal roots of 1 and thus requires more expensive complex arithmetic even where T has only real entries. We propose a simple alternative algorithm. It approximates T^{-1} by means of Discrete Sine Transform, DST, in lieu of FFT. The resulting algorithm still uses $O(K \log K)$ flops to invert T and is easily parallelizable but involves only real numbers for a real input matrix T.

2.3 Parallel Approximate Real Polynomial Division

In this chapter, we present some effective algebraic and numercical techniques for the design and analysis of algorithms and correlations between matrix and polynomial computations.

Problem of polynomial division: *Given the coefficients of two polynomials* $s(x) = \sum_{i=0}^{m} s_i x^i$, $t(x) = \sum_{i=0}^{n} t_i x^i$, *where* $s_m t_n \neq 0$, *find the coefficients of the quotient* $q(x) = \sum_{i=0}^{m-n} q_i x^i$ *and the remainder* $r(x) = \sum_{i=0}^{n-1} r_i x^i$ *of the division of* $s(x)$ *by* $t(x)$ *such that*

$$s(x) = t(x)q(x) + r(x), \quad \deg r(x) < n. \tag{3.1}$$

The problem is equivalent to the following task:

Compute $q_0, \ldots, q_{m-n}, r_0, \ldots, r_{n-1}$ such that

$$\begin{pmatrix} s_m \\ s_{m-1} \\ \vdots \\ s_n \\ s_{n-1} \\ \vdots \\ s_0 \end{pmatrix} = \begin{pmatrix} t_n & & & O \\ t_{n-1} & t_n & & \\ \vdots & \ddots & \ddots & \\ & & & t_n \\ t_0 & \ddots & \ddots & t_{n-1} \\ & \ddots & \ddots & \\ & & \ddots & \vdots \\ O & & & t_0 \end{pmatrix} \begin{pmatrix} q_{m-n} \\ q_{m-n-1} \\ \vdots \\ q_0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ r_{n-1} \\ \vdots \\ r_0 \end{pmatrix}. \tag{3.2}$$

Theorem 2.3.1 *Let T be a $K \times K$ real triangular Toeplitz matrix. Then $O(K \log K)$ real flops are needed to approximate the inverse of the triangular Toeplitz matrix T^{-1}.*

Proof: We first recall from [BC83], [KO96] that the $K \times K$ matrix $Z + Z^T$ can be diagonalized by the DST, that is, $Z + Z^T = S$ diag $(d_0, \ldots, d_{K-1})S$, where $S = [\sqrt{\frac{2}{K}}\sin\frac{\pi ij}{K}]$, $i, j = 0, \ldots, K-1$, is the matrix of Discrete Sine Transform, DST: $S^2 = I$, $d_i = 2\cos\frac{\pi i}{K}$, $S^2 = I$. Then we observe that $Z + \epsilon^2 Z^T = \epsilon D_\epsilon^{-1}(Z + Z^T)D_\epsilon$, where ϵ is any scalar and $D_\epsilon = \mathrm{diag}(1, \epsilon, \ldots, \epsilon^{K-1})$. Therefore,

$$Z + \epsilon^2 Z^T = \epsilon D_\epsilon^{-1} S \text{ diag } (d_0, \ldots, d_{K-1}) S D_\epsilon, \tag{3.3}$$

$$(Z + \epsilon^2 Z^T)^{-1} = \epsilon^{-1} D_\epsilon^{-1} S \text{ diag } (d_0^{-1}, \ldots, d_{K-1}^{-1}) S D_\epsilon. \tag{3.4}$$

Now, we approximate $T = \sum_{i=0}^{K-1} t_i Z^i$ by the matrix $T_\epsilon^* = \sum_{i=0}^{K-1} t_i (Z + \epsilon^2 Z^T)^i$ and obtain that $(T_\epsilon^*)^{-1} = \epsilon D_\epsilon^{-1} S$ diag $(\sum_{i=0}^{K-1} t_i (d_j)^i)^{-1} S D_\epsilon$, $j = 0, \ldots, K-1$. Clearly, $(T_\epsilon^*)^{-1} \to T^{-1}$ as $\epsilon \to 0$. Therefore, for a positive ϵ approximating 0, we approximate T^{-1} in $O(K \log K)$ flops involving only real values if T is real. The same goal can be achieved by means of any of the 4 known variants of the DST and 4 known variants of DCT (Discrete Cosine Transform) (cf. [KO96]). Each of the 8 transforms diagonalizes a tridiagonal matrix H with the triple of the central diagonals of the form:

$$a, 1, \ldots 1, b$$
$$c, 0, 0, \ldots 0, 0, d$$
$$a, 1, \ldots 1, b$$

where $c = d = 0$ unless $a = b = 1$. We write $D(u, v, w) = \text{diag}\,(u, v, v, \ldots, v, v, w)$. If $c = d = 0$, we have $H = D(a, 1, b)(Z + Z^T)D(a, 1, b)$ [KO96], and we approximate Z by $Z + \epsilon^2 Z^T = \epsilon D_\epsilon^{-1} D^{-1}(a, 1, b) H D^{-1}(a, 1, b) D_\epsilon$. If $a = b = 1$, we have $H = Z + Z^T + D(c, 0, d)$ [KO96], and we approximate Z by the matrix $\epsilon D_\epsilon^{-1} H D_\epsilon = Z + \epsilon D(c, 0, d) + \epsilon^2 Z^T$. In both cases, the matrix aproximating Z is diagonalized as soon as H is diagonalized by DST or DCT, and then the algorithm for the approximation of T^{-1} is immediatley extended. $\square$

Chapter 3

The Complexity of the Algebraic Eigenproblem

3.1 Summary

The eigenproblem for an n-by-n matrix A is the problem of the approximation (within a relative error bound 2^{-b}) of all the eigenvalues of the matrix A and computing the associated eigenspaces of all these eigenvalues. The arithmetic complexity of this problem is bounded by $O(n^3 + (n \log^2 n) \log b)$. If the characteristic and minimum polynomials of the matrix A coincide with each other (which is the case for generic matrices of all classes of general and special matrices that we consider), then the latter deterministic cost bound can be replaced by the randomized bound $O(K_A(2n) + n^2 + (n \log^2 n) \log b)$ where $K_A(2n)$ denotes the cost of the computation of the $2n - 1$ vectors $A^i\mathbf{v}$, $i = 1, \ldots, 2n-1$, maximized over all n-dimensional vectors $\mathbf{v}$; $K_A(2n) = O(M(n) \log n)$, for $M(n) = o(n^{2.376})$ denoting the arithmetic complexity of $n \times n$ matrix multiplication. In particular, this bound turns into $O(n^2 \log n + (n \log^2 n) \log b)$ for generic structured matrices of the classes of $n \times n$ Toeplitz, Hankel, Toeplitz-like, Hankel-like and Toeplitz-like-plus-Hankel-like matrices. This bound is optimal (up to a logarithmic factor) for each of the latter classes of input matrices. We also prove similar nearly optimal upper bounds for the generic Cauchy-like, Vandermonde-like and sparse matrices.

3.2 Introduction

3.2.1 Matrix eigenproblem and some open issues

The solution of the matrix eigenproblem (that is, the problem of the computation of the matrix eigenvalues and the eigenspaces of the associated eigenvectors) is a central classical topic of applied linear algebra. The history of this problem can be traced back at least to the first half of the 19th century, when, motivated by applications to celestial mechanics, Cauchy discovered his celebrated Interlacing Theorem for the eigenvalues. Recently the eigenproblem turned out to be highly important also for the solution of polynomial systems of equations [AuSt88], [Ste96], [MP98], [BMP98].

The subject has enormous bibliography (see e.g. [GL96] and references therein) and causes never stopping stream of research articles in the leading journals on numerical analysis and applied linear algebra. The importance of this topic was highly recognized by numerical analysts (e.g. works on it have been awarded by the SIAM Prize in Applied Linear Algebra) but much less so by the computer science theoreticians, even though some major practical and theoretical algorithmic problems in this area were widely open.

In particular, the customary iterative algorithms for the eigenproblem are relatively slow for the worst case (unsymmetric) input matrix. We will state this quantitatively, measuring the arithmetic computational cost by the number of ops involved, where "ops" will be our abbreviation for both arithmetic operations (over real or complex numbers) and comparisons (of the absolute values or the moduli of pairs of real or complex numbers). We observe that for a general $n \times n$ input matrix A, the most popular QR algorithm requires order of n^3 ops to reduce the matrix to Hessenberg form and then order of Mn^3 ops to approximate all the real and complex eigenvalues of A, where M, the average number of iterations per eigenvalue, is unbounded both theoretically and practically. Other known practical iterative algorithms have no better computational cost estimates and no better performance [GL96]. All of them routinely fail in practice already for n of order 50 in the highly important case where some eigenvalues are multiple or form clusters. This observation is stated explicitly e.g. in [Go94], pp. 1062-1063, whereas the solution in the case of a much higher order of n and in the presence of clustered eigenvalues is usually required, say, in the cited applications to polynomial systems of equations.

The computational complexity estimates based on the customary algorithms are not satisfactory even for the much simpler *eigenspace problem*, which is a subproblem of the eigenproblem. Namely, in the eigenspace problem, one assumes all the eigenvalues of an $n \times n$ matrix available and seeks the eigenspaces of the eigenvectors associated with these eigenvalues. The solution method of practical choice is the Inverse Power Iteration, but it supports no finite complexity estimates for the worst case input (see Appendix B).

Similar deficiency is observed for the eigenproblem for the various highly important classes of special (sparse and dense structured, e.g. Toeplitz-like and Cauchy-like) matrices, for which one should expect substantial acceleration of the solution by exploiting the sparsity and/or structure. Here again, the full complexity analysis of the customary solution algorithms is missing.

The above background should have made the eigenproblem a major computer science subject in the areas of computational complexity and the design and analysis of algebraic algorithms. So far the situation is quite opposite, however, apart from the estimates for the complexity of some selected algorithms [KW92], the solution of the (simpler) symmetric tridiagonal eigenvalue problem in [BP91], [BP98] and some treatment of the sparse eigenvalue problem in [R95] (cf. Remark A.1 in our Appendix A).

In this chapter, we will study the arithmetic complexity issues for the eigenproblem for both general and special $n \times n$ input matrices, and will arrive at tight estimates (up to polylogarithmic factors).

3.2.2 The arithmetic complexity of the eigenproblem for general matrices

Let us next estimate the arithmetic complexity of the eigenproblem for general $n \times n$ matrices. To solve the eigenproblem with controlled computational cost for an input matrix A possibly having clustered and/or multiple eigenvalues, we rely on the well-known observation that all the eigenvalues of an $n \times n$ matrix A are the zeros of its characteristic polynomial $c_A(x)$. This observation immediately enables us to reduce the solution of the eigenproblem to the following stages:

a) the reduction of the eigenproblem for a given matrix to the one for a matrix in a canonical form (that is, for a Hessenberg matrix or, more specially, for a Frobenius matrix, for a block triangular or block diagonal matrix with Frobenius diagonal blocks, or for a tridiagonal matrix); as a by-product of this stage, the coefficients of the characteristic polynomial $c_A(x)$ of the original input matrix A are output or become available readily, in $O(n)$ additional ops,

b) the approximation of the eigenvalues of A as the zeros of $c_A(x)$, and

c) the computation of the eigenspaces of the approximated eigenvalues, by exploiting the available reduction of A to the canonical form.

Stage a) has been thoroughly investigated already several decades ago [FF63]. Several well-known algorithms reduce a generic $n \times n$ matrix A to a Frobenius matrix, and we also prove in section 3.10 that an unsymmetric modification of Lanczos algorithm reduces it to the tridiagonal form. (Generic $n \times n$ matrices cover all $n \times n$ matrices but ones forming an algebraic variety of a lower dimension.) Furthermore, the algorithms for Hessenberg reduction are well known [GL96], [BP94], and by combining and extending some known techniques, Keller-Gehrig in [K-G85] devised a solution algorithm that reduced any $n \times n$ input matrix A to the triangular Frobenius form. All these algorithms are performed at nearly optimal arithmetic cost of $O(M(n) \log n)$ ops, whereas $\Omega(M(n))$ is a lower bound on the arithmetic complexity of the computation of $c_A(x)$ for a general input matrix (see [BP94], exercise 2.6 on p.213). Here and hereafter, $M(n)$ denotes the number of ops required for $n \times n$ matrix multiplication: $M(n) = O(n^{\omega})$ where $\omega < 2.376$ in theory, and ω ranges from $\log_2 7 < 2.808$ to 3 in practical algorithms (cf. [BP94], p. 94; [GL96], [Hig96], [K,a]).

At stage b) we fix some matrix norm $||\cdot||$ and absolute output error bound $\epsilon = 2^{-b}||A||$ and apply the recent nearly optimal polynomial rootfinders (see section 3.11) to approximate all the eigenvalues of A within ϵ in arithmetic time

$$t(n,b) = O((n \log^2 n)(\log b + \log^2 n)), \tag{2.1}$$

whereas n and $\Omega(\log b)$ are the known lower bounds on the complexity of polynomial rootfinding (cf. [R87] and [P97]).

For an $n \times n$ matrix A given with its eigenvalue λ, stage c) amounts to the computation of the null space of the matrix $\lambda I_n - A$, I_n being the $n \times n$ identity matrix. The known algorithms compute the null space in $O(M(n))$ ops (see [BP94], pp. 109-110), which gives us the bound of $O(nM(n)) = O(n^{\omega+1})$ ops for the entire eigenspace problem. Such a bound for the eigenspace problem also dominates the overall complexity of the eigenproblem. Our study in section 3.12, however, shows that $O(n^3)$ ops are sufficient to solve the eigenspace problem for an $n \times n$ matrix A given in the Hessenberg form or in the more special triangular Frobenius form whereas the cost decreases to $O(n^2)$ ops if A is a tridiagonal or Frobenius matrix or is represented in the diagonal Frobenius form. (The latter result was known in the Frobenius and block diagonal Frobenius cases.) Summarizing, we obtain the following results:

Theorem 3.2.1 *The deterministic arithmetic complexity of the eigenproblem for any $n \times n$ matrix A is bounded by $O(n^3) + t(n, b)$ ops for $t(n, b)$ of (2.1) and for $2^{-b}||A||$ denoting the required upper bound on the absolute output error of the approximation of the eigenvalues of A where $||\cdot||$ denotes any fixed matrix norm. For generic $n \times n$ matrix A, the complexity is bounded by $O(M(n)\log n) + t(n, b)$ ops, where $M(n)$ denotes the complexity of $n \times n$ matrix multiplication, $M(n) = o(n^{2.376})$.*

Remark 3.2.1 *Clearly, the term $t(n, b)$ can be deleted from the former of the above estimates if $\log b = O(n^2/\log^2 n)$, that is, if $b = 2^{O(n^2/\log^2 n)}$, and from the latter of the estimates if $\log b = O(M(n)/(n \log n))$, that is, if $b = 2^{O(M(n)/(n\log n))}$.*

3.2.3 The arithmetic complexity of the eigenproblem for sparse and dense structured matrices

As we will see in section 3.12, stage c) of the solution of the eigenproblem is much simplified for any $n \times n$ matrix A that satisfies the equation

$$m_A(x) = c_A(x), \tag{2.2}$$

where $m_A(x)$ denotes the minimum polynomial of A. Equation (2.2) holds generically for an $n \times n$ input matrix A, and if (2.2) holds, then stage a) becomes the bottleneck, that is, the cost of the computations at stage a) dominates the overall computational cost of the solution of the eigenproblem. In section 3.5, we will show that the cost can be decreased dramatically in the case where the input matrix A is special, that is, sparse or dense structured.

Sparse matrices A are characterized by the patterns of their nonzero entries, whose total number will be denoted by f_A. It is realistic to assume that

$$f_A = O(n^\beta), \tag{2.3}$$

with $\beta < 2$ or even $\beta = 1$ for an $n \times n$ sparse matrix A.

There are several important families of dense structured matrices, the most celebrated being the classes of *Toeplitz, Hankel, Vandermonde and Cauchy matrices,* which we display in the next table.

Table 1. Dense Structured matrices.

Toeplitz matrices: $T = (t_{i-j})_{i,j=0}^{n-1}$	Hankel matrices: $H = (h_{i+j})_{i,j=0}^{n-1}$.
Vandermonde matrices: $V = (x_i^j)_{i,j=0}^{n-1}$	Cauchy matrices: $C = (\frac{1}{x_i - y_j})_{i,j=0}^{n-1}$

For the latter classical matrices, important extensions were more recently developed called *Toeplitz-like, Hankel-like, Vandermonde-like and Cauchy-like matrices* in [BP94]. In some applications, one deals with *Toeplitz-like + Hankel-like matrices* (obtained as the sum of pairs of Toeplitz-like and Hankel-like matrices and thus extending both of these two classes).

It is a basic property of sparse and dense structured matrices that they can be multiplied by vectors fast. Formally, let v_A denote the number of ops required to multiply a given $n \times n$ matrix A by a vector $\mathbf{v}$ and let $K_A(h)$ denote the number of ops required to compute the $n \times h$ Krylov matrix, $K(A, \mathbf{v}, h) = (A^i \mathbf{v})_{i=0,\dots,h-1}$, where $\mathbf{v}$ is a generic n-dimensional vector. We have

$$K_A(h) \leq (h-1)v_A, \tag{2.4}$$

$$v_A = 2n^2 - n, \ K_A(h) \leq (2n-1)(h-1)n, \tag{2.5}$$

$$v_A \leq 2f_A, \ K_A(h) \leq 2(h-1)f_A \tag{2.6}$$

for any $n \times n$ matrix A (where f_A is the number of nonzero entries of A),

$$v_A = O(n \log n), \ K_A(h) = O(hn \log n) \tag{2.7}$$

for any $n \times n$ Toeplitz, Hankel, Toeplitz-like, Hankel-like, or Toeplitz-like+Hankel-like matrix A, and

$$v_A = O(n \log^2 n), \ K_A(h) = O(hn \log^2 n), \tag{2.8}$$

for any $n \times n$ Vandermonde, Vandermonde-like, Cauchy, or Cauchy-like matrix A (cf. [BP94]).

We will prove that for every considered class of special matrices (which is either a linear subspace or an algebraic variety in the linear space of dimension n^2 of the entries of all $n \times n$ matrices) equation (2.2) holds for generic matrices of this class (that is, for all matrices of this class except for ones that form an algebraic variety of a lower dimension).

We will next state (in terms of n, b and $K_A(n)$) our estimates for the complexity of the eigenproblems of $n \times n$ sparse and dense structured matrices A satisfying (2.2).

Theorem 3.2.2 *If an $n \times n$ matrix A satisfies (2.2), then its eigenproblem can be solved by means of generating $4n-2$ random parameters and then performing $t(n,b) + O(n^2 + K_A(n))$ ops for $t(n,b)$ of (2.1) and for $2^{-b}||A||$ denoting the required upper bound on the output errors for the eigenvalues, where $||\cdot||$ denotes any fixed matrix norm. The cost* *bound does not include the cost of the generation of random parameters. Assuming that these parameters are sampled from a fixed finite set S of cardinality $|S|$ independently of each other under the uniform probability distribution on S, the algorithm supporting the above arithmetic complexity estimate either outputs FAILURE or otherwise, with a probability at least $(1-(n+1)n/(2|S|))(1-2n/|S|)$, produces correct output for a matrix A satisfying (2.2). The algorithm can be applied to any $n \times n$ matrix A and outputs FAILURE unless (2.2) holds.*

Remark 3.2.2 *The term $t(n,b)$ can be deleted from the above complexity estimate if $\log b = O(n/\log^2 n)$, that is, if $b = 2^{O(n/\log^2 n)}$. The term n^2 can be deleted unless $K_A(n) = o(n^2)$.*

If we rely on bounds (2.4) and (2.5), Theorem 3.2.2 gives no improvement of the complexity estimates of Theorem 3.2.1. For sparse and dense structured matrices, we immediately obtain a major improvement based on (2.6) –(2.8).

Corollary 3.2.1 *The complexity estimate of Theorem 3.2.2 turns into $t(n,b)+O(n^2+nf_A)$ for $n \times n$ sparse matrices A having f_A nonzero entries (which gives us the bound $t(n,b)+O(n^2)$ if $f_A = O(n)$); the estimate turns into $t(n,b)+O(n^2 \log n)$ for $n \times n$ Toeplitz-like+Hankel-like matrices A, and into $t(n,b)+O(n^2 \log^2 n)$ for Cauchy-like and Vandermonde-like matrices A.*

Remark 3.2.3 *The latter bounds are optimal (up to polylogarithmic factors) because n^2 ops are required already to output the n^2 coordinates of the n eigenvectors of an $n \times n$ matrix.*

For comparison, the knows estimates for the complexity of the eigenproblem of the above highly important classes of special matrices do not improve the bound $O(M(n))$=$O(n^\omega)$ known for general matrices satisfying (2.2).

3.2.4 Organization of the chapter

We organize this chapter as follows. In the next section we introduce the matrix eigenproblem. In sections 3.4 and 3.5 we study its reduction to canonical forms together with the computation of the characteristic polynomial, $c_A(x)$. In sections 3.6–3.8, we define various classes of dense structured matrices and prove that equation (2.2) holds generically for the matrices of these classes as well as for sparse matrices. In section 3.9, we cover fast randomized solution of singular but consistent Toeplitz or Toeplitz-like linear systems of equations (over any field of constants), which is required in section 3.5. In section 3.10, we study the reduction of a matrix to the tridiagonal form by means of an unsymmetric variant of Lanczos algorithm and prove that this works for generic input. In section 3.11, we cover approximation of the eigenvalues as the zeros of $c_A(x)$ and the computation of their algebraic multiplicities. In section 3.12, we compute the eigenspaces of all the eigenvalues. In section 3.13, we briefly discuss some open issues, in particular the solution of the eigenproblem for the special matrices associated with polynomial systems and the extension of our arithmetic complexity results to the bit-complexity estimates. In Appendix A we recall some fast deterministic algorithms for the computation of the characteristic polynomial where A is sparse or is dense and structured. In Appendix B we comment on the Inverse Power Iteration for the computation of the eigenvectors.

3.3 Matrix Eigenproblem

In this section we recall some definitions and basic results (cf. [FF63], [GL96], [Wil65]).

Definition 3.3.1 *For an $n \times n$ matrix A, let*

$$AV = V\Lambda, \tag{3.1}$$

where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_q)$ is a $q \times q$ diagonal matrix, $V = (\mathbf{v}_1, \ldots, \mathbf{v}_q)$ is an $n \times q$ matrix of full rank with columns $\mathbf{v}_1, \ldots, \mathbf{v}_q$, and q is maximum, $q \le n$. Then every pair $(\lambda_j, \mathbf{v}_j)$ is the pair of the eigenvalue *λ_j and the* eigenvector *$\mathbf{v}_j$ of A associated to each other, and the computation of such pairs for $j = 1, \ldots, q$ is the* eigenproblem *for A. $\lambda_1, \ldots, \lambda_q$ must not be all distinct. The number μ_j of times the eigenvalue λ_j is repeated in Λ is called the* geometric multiplicity *of λ_j. λ_j is* geometrically simple *if $\mu_j = 1$. μ_j is the dimension of the* eigenspace of the eigenvalue *λ_j, defined as the linear space of the eigenvectors associated to this eigenvalue.*

Remark 3.3.1 *One may scale the eigenvalues of A by scaling A because (3.1) implies that $hAV = V(h\Lambda)$ for any scalar h.*

Definition 3.3.2 *Hereafter, I_k denotes the $k \times k$* identity matrix, 0 *denotes a* null matrix *of appropriate size,* $\det W$ *denotes the* determinant *of a matrix W; W^T and W^H denote the* transpose *and the* Hermitian transpose *of a matrix or a vector W, respectively.*

Definition 3.3.3 $c_A(x)$=$\det(xI_n - A)$ *is* the characteristic polynomial *of A. $m_A(x)$, the* minimum polynomial *of A, is the monic polynomial of the minimum degree such that $m_A(A) = 0$.*

Theorem 3.3.1 $c_A(A) = 0$. *Furthermore, $m_A(x)$ divides $c_A(x)$; $m_A(\lambda_j) = c_A(\lambda_j) = 0$ if and only if λ_j is an eigenvalue of A.*

Definition 3.3.4 μ_j^+, *the* algebraic multiplicity *of an eigenvalue λ_j of A, is the multiplicity of the zero λ_j of $c_A(x)$, that is, $\mu_j^+ = \max\{d : (x - \lambda_j)^d$ divides $c_A(x)\}$.*

We have

$$\mu_j^+ \geq \mu_j \text{ for all } j. \tag{3.2}$$

Theorem 3.3.2 *Let (3.1) hold, let T be an $n \times n$ nonsingular matrix, and let*

$$\hat{A} = TAT^{-1}. \tag{3.3}$$

Then $\hat{A}(TV) = (TV)\Lambda$.

Definition 3.3.5 *The matrix transformation (3.3) of A into $\hat{A}$ is called* similarity transformation, *which reduces the eigenproblem for A to one for $\hat{A}$ and is said to* reduce A *to $\hat{A}$.*

The initial customary step of the solution of the eigenproblem for a given matrix A is the reduction of A (by a similarity transformation) to some canonical form for which the solution of the eigenproblem is simpler. The next two definitions describe some of these canonical forms of matrices.

Definition 3.3.6 *$A = (a_{i,j})$ is a* tridiagonal matrix *if $a_{i,j} = 0$ for $|i - j| > 1$. More generally, A is a* Hessenberg matrix *if $a_{i,j} = 0$ for $i - j > 1$. A Hessenberg or tridiagonal matrix $(a_{i,j})$ is* unreduced *if $a_{i+1,i} \neq 0$ for all i.*

Remark 3.3.2 *Actually, we narrowed the class of Hessenberg matrices to the upper Hessenberg matrices and similarly for the matrices in the triangular Frobenius form defined below (in Definition 3.3.7) because this is sufficient for our eigenstudy.*

The following simple observations specify the structure of Hessenberg and tridiagonal matrices.

Theorem 3.3.3 *Every Hessenberg matrix A can be represented as a block triangular matrix with unreduced Hessenberg diagonal blocks. If A is tridiagonal, then so are these blocks too, and each superdiagonal block either vanishes or has a single nonzero entry, located in its southwestern corner.*

The next special case of Hessenberg matrices will be important for our study.

Definition 3.3.7 *A special $n \times n$ Hessenberg matrix of the form*

$$F = F(p) = \begin{pmatrix} 0 & \cdots & & 0 & -p_0 \\ 1 & \ddots & & & \vdots \\ & \ddots & & & \vdots \\ & & \ddots & 0 & \vdots \\ 0 & & & 1 & -p_{n-1} \end{pmatrix}$$

is called a Frobenius (companion) matrix. *A Hessenberg matrix A is* in the (block) triangular Frobenius form *if this is a block upper triangular matrix,*

$$A = \begin{pmatrix} F_1 & & * \\ & \ddots & \\ & & \ddots & \\ 0 & & & F_k \end{pmatrix}, \tag{3.4}$$

whose diagonal blocks $F_1, \ldots, F_k$ are Frobenius matrices. Here, all the subdiagonal blocks are null matrices, and $*$ stays for the superdiagonal blocks. If all of them are null matrices, then the matrix is in the (block) diagonal Frobenius form.

Remark 3.3.3 *A Frobenius matrix F is available as soon as its characteristic polynomial $c_F(x) = x^n + \sum_{n=0}^{n-1} p_i x^i$ is available and vice versa.*

Remark 3.3.4 *For a matrix A of (3.4), we have*

$$c_A(x) = \prod_{i=1}^{k} c_{F_i}(x). \tag{3.5}$$

Hereafter, we will assume computations with real or complex numbers, will measure the computational cost by the number of arithmetic operations and comparisons involved, and will refer to all such operations as *ops*.

By applying Gaussian elimination with row interchange, we obtain the following results.

Theorem 3.3.4 *$O(n^2)$ ops suffice to factorize an $n \times n$ nonsingular Hessenberg matrix A as a product PLU where P is a permutation matrix, L and U are lower and upper triangular matrices, respectively. The cost bound decreases to $O(n)$ either if A is tridiagonal or if $A = \lambda I_n - F$ for a scalar λ and a matrix F given in the diagonal Frobenius form. If the PLU factorization of A has been computed, then $O(n)$ additional ops are sufficient to compute $\det A$ and to decide whether A is singular or not.*

Corollary 3.3.1 *$O(n^2)$ ops suffice to solve a nonsingular linear system of equations with a Hessenberg coefficient matrix A. The cost bound decreases to $O(n)$ if A is a tridiagonal matrix or equals $\lambda I_n - F$ where F is a matrix in the diagonal Frobenius form and λ is a scalar.*

3.4 Reduction of a Matrix to the Hessenberg, Triangular and Diagonal Frobenius Forms

We will start this section with some definitions partly repeating ones of the introduction.

Definition 3.4.1 *Hereafter, $M(n)$ and $I(n)$ denote the arithmetic cost of $n \times n$ matrix multiplication and inversion, respectively.*

It is well-known (see e.g. [BP94], p. 213) that

$$M(n) = O(n^\omega),\ 2 \le \omega < 2.376, \tag{4.1}$$

$$M(n) = O(I(n)),\ I(n) = O(M(n)).$$

Remark 3.4.1 *(cf. [BP94], p. 94; [K,a]). The asymptotic upper bound of (4.1) hides a huge overhead constant in the "O" notation. Practical customary algorithms support the bounds $M(n) = O(n^{2.808})$ and $I(n) = O(n^3)$ where the overhead constants are quite small. Estimating the computational cost in terms of $M(n)$, however, has some practical justification because matrix multiplications and the computations intensively involving them are performed very effectively on modern supercomputers [GL96], [Q94].*

Theorem 3.4.1 *Given an $n \times n$ matrix A, it suffices to use $O(M(n) \log n)$ ops in order to compute a nonsingular matrix W, its inverse W^{-1}, and the matrix WAW^{-1} in the triangular Frobenius form.*

Proof. See [K-G85]. □

Keller-Gehrig's algorithm of [K-G85], supporting Theorem 3.4.1, involves several steps of modified block Gaussian elimination, which makes it generally ineffective for sparse and/or structured matrices A.

We also recall the following results.

Theorem 3.4.2 *(see [GL96], p.314). For a pair of $n \times n$ matrices $A = \begin{pmatrix} B & C \\ 0 & G \end{pmatrix}$ and $W = \begin{pmatrix} I_\rho & X \\ 0 & I_{n-\rho} \end{pmatrix}$, where B is a $\rho \times \rho$ matrix, we have $W^{-1}AW = \begin{pmatrix} B & C - XG + BX \\ 0 & G \end{pmatrix}$.*

Remark 3.4.2 *Instead of triangular Frobenius form, one may reduce A to the more general Hessenberg form. Theorem 3.4.1 applies, of course, but other algorithms work too, and some of them are more attractive for practical computations [GL96].*

Theorem 3.4.3 *[GIKT95], pp. 383-384.) Under the assumptions of Theorem 3.4.2, let B be a Frobenius matrix. Then there exists a matrix X satisfying $XG - BX = C$.*

Based on Theorems 3.4.2 and 3.4.3, we may reduce an $n \times n$ matrix A from the triangular Frobenius to the diagonal Frobenius form. The estimated cost of such a reduction is dominated by the cost estimates for the stage of the computation of the matrices X of these theorems. The latter cost estimates generally exceed $M(n)$ by order of magnitude, even where B and G are Frobenius matrices.

3.5 Randomized Reduction to a Frobenius Matrix

We will again start this section with definitions, partly repeating ones of the introduction.

Definition 3.5.1 *For an $n \times n$ matrix A and an n-dimensional vector $\mathbf{v}$, the $n \times n$ matrices $K(A, \mathbf{v}, h) = (A^i \mathbf{v})_{i=0,\dots,h-1}$ for $h = 1, 2, \dots,$ are called* Krylov matrices *induced by A and $\mathbf{v}$.*

Definition 3.5.2 *For an $n \times n$ matrix A and a positive integer h, let $K_A(h)$ denote the arithmetic complexity of the computation of the Krylov matrix $K(A, \mathbf{v}, h)$ maximized over all n-dimensional vectors $\mathbf{v}$.*

Definition 3.5.3 *A subset W of a linear space V or, more generally, of an algebraic variety V of a dimension k is algebraically open or* generic *if $V - W$ is an algebraic variety of dimension less than k. A property holds* generically *for a fixed class of matrices (or, equivalently, for a* generic *matrix of a fixed class) if it holds generically for the matrices of this class.*

Definition 3.5.4 Uniform sampling *of h elements from a finite set S is their selection from S independently of each other under the uniform probability distribution on S. $|S|$ is the cardinality of a set S.*

We have the following well-known results (cf. e.g [Wi86], [KP91], [BP94] on Theorem 4.1 and [GL96], pp. 348-349, on Theorem 4.2).

Theorem 3.5.1 *$m_A(x) = c_A(x)$ (that is, equation (2.2) holds) for an $n \times n$ matrix A if and only if*

$$\text{rank } (K(A, \mathbf{v}, n)) = n \tag{5.1}$$

or, equivalently, if and only if the $n \times n$ matrix $H_n = (\mathbf{u}^T A^{i+j} \mathbf{v})_{i,j=0}^{n-1}$ is nonsingular where the vectors $\mathbf{u}$ and $\mathbf{v}$ (both of dimension n) have $2n$ indeterminates as their coordinates.

Corollary 3.5.1 *Equation (5.1) (for a fixed vector $\mathbf{v}$) holds generically for $n \times n$ matrices. Equation (2.2) holds generically for $n \times n$ matrices A.*

Remark 3.5.1 *Corollary 3.5.1 can be deduced alternatively from the well-known fact that equation (2.2) holds if an $n \times n$ matrix A has n distinct eigenvalues and from Theorem 3.3.1.*

Theorem 3.5.2 *Let A be an $n \times n$ matrix. Let $\mathbf{v}$ be a vector of dimension n. Let equations (2.2), (5.1) hold. Then*

$$F = K^{-1}AK, \ \textit{for} \ K = K(A, \mathbf{v}, n), \tag{5.2}$$

is the $n \times n$ Frobenius matrix satisfying the equation

$$c_F(x) = c_A(x). \tag{5.3}$$

Theorem 3.5.2 for $\mathbf{v} = (1, 0, \ldots, 0)^T$ leads to the Danilevski algorithm which under (5.1) computes a similarity transformation of a matrix A into the Frobenius matrix F and outputs $c_F(x) = c_A(x)$ and the matrix K of (5.2). The theorem has the following corollary, involving $M(n), I(n)$ and $K_A(n)$ of Definitions 3.4.1 and 3.5.2.

Corollary 3.5.2 *Let an $n \times n$ matrix A and a vector $\mathbf{v}$ satisfy equations (2.2), (5.1). Then the Frobenius matrix F and the Krylov matrix $K = K(A, \mathbf{v}, n)$ satisfying equation (5.2) can be computed at the cost $K_A(n) + 2M(n) + I(n)$.*

Now we observe that for the solution of the eigenproblem we need only the matrices F and K of (5.2) but not K^{-1}. This motivates our next goal to remove the terms $2M(n)$ and $I(n)$ from the cost bound of Corollary 3.5.1. Such a removal will dramatically accelerate the computations in the case where A is a sparse and/or structured matrix (cf. (2.6)-(2.8)). Towards this goal, we first recall the next theorem, fundamental for the study of randomized algebraic algorithms. The theorem is due to [DL78] but is more widely known from the papers [Z79] and [S80].

Theorem 3.5.3 *Let $p(x) = p(x_1, \ldots, x_m)$ be a nonzero m-variate polynomial over and field or ring R. Let $p(x)$ have total degree d. Let $x_1^* \ldots, x_m^*$ be uniformly sampled from a fixed finite subset S of R. Then $p(x_1^* \ldots, x_m^*) \neq 0$ with a probability at least $1 - d/|S|$.*

The algorithm supporting the next theorem probabilistically computes the minimum polynomial $m_A(x)$ of a given matrix A.

Theorem 3.5.4 *Let A be a fixed $n \times n$ matrix, let S be a fixed finite set and let $\mathbf{v}$ be a random vector of dimension n with its entries uniformly sampled from the set S. Then it suffices to sample uniformly $3n-2$ additional random values from S and to perform $K_A(2n) + O(n^2)$ ops in order to compute the matrix $K(A, \mathbf{v}, n)$ and a monic divisor $d(x)$ of the polynomial $m_A(x)$ such that $d(x) = m_A(x)$ with a probability at least $(1 - \frac{(n+1)n}{2|S|})(1 - \frac{2n}{|S|})$.*

Proof. The modification by [KP91] of the algorithm of [Wi86] uniformly samples $2n$ random parameters from a finite set S, to define the $2n$ components of two vectors $\mathbf{u}$ and $\mathbf{v}$, each of dimension n. Then (cf. [KP91]), at the cost of performing $K_A(2n) + O(n^2)$ ops, the $n \times (2n)$ matrix $K(A, \mathbf{v}, 2n)$ and the vector $\mathbf{h} = \mathbf{u}^T K(A, \mathbf{v}, n) = (\mathbf{u}^T A^i \mathbf{v})_{i=0}^{2n-1}$ are computed, which defines the $k \times k$ Hankel matrices,

$$H_k = (\mathbf{u}^T A^{i+j} \mathbf{v})_{i,j=0}^{k-1}, \quad k = 1, \ldots, n, \tag{5.4}$$

satisfying

$$\det H_\rho \neq 0, \text{ for } \rho = \operatorname{rank}(H_n). \tag{5.5}$$

The coefficient vector $\mathbf{d} = (d_i)_{i=0}^{r-1}$ of the polynomial $d(x) = \sum_{i=0}^{r-1} d_i x^i$ is computed as the solution of the following nonsingular linear system of ρ equations:

$$H_\rho \mathbf{d} = (\mathbf{u}^T A^i \mathbf{v})_{i=\rho}^{2\rho-1}. \tag{5.6}$$

It is immediately observed that the polynomial $d(x)$ divides $m_A(x)$; furthermore, it is proved in [KP91] (based on Theorem 3.5.3) that $d(x)$ equals the minimum polynomial $m_A(x)$ with a probability at least $1 - \frac{2n}{|S|}$. It remains to compute ρ and to solve the linear system (5.6). This task will be solved in section 3.9, where we will recall and slightly elaborate further the algorithm of [Kal95]. The solution requires to sample $2n-2$ additional random values uniformly from the set S and in addition to perform $O(n \log^2 n)$ ops. This will complete the proof of Theorem 3.5.4. □

Now, for an $n \times n$ matrix A we immediately devise the following randomized algorithm, which tests if (5.1) holds and, if so, computes two matrices, F (Frobenius) and K (Krylov) of (5.2), (5.3), and also outputs $c_A(x)$ (cf. Remark 3.3.3).

Algorithm 3.5.1 *(reduction of a generic matrix to the Frobenius matrix).*

Input: an $n \times n$ matrix A and a finite set S.

Output: FAILURE or an $n \times n$ nonsingular matrix K and the Frobenius matrix F satisfying equations (5.2), (5.3), as well as the characteristic polynomial $c_A(x) = c_F(x)$.

Computations:

1. Apply the randomized algorithm supporting Theorem 3.5.4 to compute a matrix $K = K(A, \mathbf{v}, n)$ (for a random vector $\mathbf{v}$) and a divisor $d(x)$ of $m_A(x)$.

2. If the degree of $d(x)$ is n, output $d(x)$, the matrix K and the Frobenius matrix F satisfying $c_F(x) = d(x)$. Otherwise output FAILURE.

Correctness of the algorithm follows from Theorems 3.5.2 and 3.5.4.

By summarizing the estimates for the computational cost and the failure probability of the algorithm, we obtain the following result.

Theorem 3.5.5 *If Algorithm 3.4.1 is applied to an $n\times n$ matrix A not satisfying (2.2), it always outputs FAILURE. If the algorithm is applied to an $n\times n$ matrix A satisfying (2.2), then it either outputs FAILURE or otherwise, with a probability at least $(1-\frac{(n+1)n}{2|S|})(1-\frac{2n}{|S|})$, correctly computes a pair of $n \times n$ matrices K and F (K a nonsingular Krylov matrix, F a Frobenius matrix) satisfying (5.2), (5.3), so that F defines the characteristic polynomial $c_A(x) = c_F(x)$. The algorithm samples uniformly $4n-2$ random values from S and then performs $K_A(2n)+O(n^2)$ ops (not including the cost of the random sampling).*

Remark 3.5.2 *As soon as the matrices K and F of (5.2) are available, Theorem 3.3.2 reduces the eigenproblem of A to the one of F.*

Remark 3.5.3 *The complexity bound of Theorem 3.5.5 is smaller than one of Theorem 3.4.1 by order of magnitude if the input matrix A is sparse and/or structured (cf. (2.6)–(2.8)).*

3.6 Equation (2.2) for General, Vandermonde-like, Cauchy-like, and Sparse Matrices

By Corollary 3.5.1, equation (2.2) holds generically for $n \times n$ matrices and also for the matrices of any subspace or algebraic variety in the space of $n \times n$ matrices provided that it holds for at least one matrix A in this subspace or variety. To prove the existence of such a matrix A, we will use, in particular, the following definition and result.

Definition 3.6.1 *An $l \times l$* leading principal *or* northwestern *submatrix of an $n \times n$ matrix W is the submatrix formed by the entries lying in the first l columns and first l rows of W, for $l \leq n$. If the $k \times k$ leading principal (northwestern) submatrices of a matrix W of rank r are nonsingular for $k = 1, \ldots, r$, then W is said to have* generic rank profile. *A matrix is* strongly nonsingular *if it is nonsingular and has generic rank profile.*

Theorem 3.6.1 *[Wi86]. Equation (2.2) holds for an $n \times n$ matrix $A = WY$, provided that Y is the $n \times n$ diagonal matrix with indeterminates filling its diagonal and that the matrix W has generic rank profile.*

Let us next show that equation (2.2) holds generically for Vandermonde-like matrices and Cauchy-like matrices. We will first recall their definitions (cf. [BP94], [GO94]).

Definition 3.6.2 *Let $x_0, \ldots, x_{n-1}$ denote n distinct values and let $s_0, t_0, \ldots, s_{n-1}, t_{n-1}$ denote $2n$ distinct values. Let $\mathbf{x} = (x_i)_{i=0}^{n-1}$, $\mathbf{s} = (s_i)_{i=0}^{n-1}$, $\mathbf{t} = (t_j)_{j=0}^{n-1}$. Then the $n \times n$ matrices $V = V(\mathbf{x}) = (x_i^j)_{i,j=0}^{n-1}$ and $C = C(\mathbf{s}, \mathbf{t}) = (\frac{1}{s_i - t_j})_{i,j=0}^{n-1}$ are called* Vandermonde *and* Cauchy matrices, *respectively. Furthermore, the $n \times n$ matrices $V_f(\mathbf{x}, G, H)$ and $C(\mathbf{s}, \mathbf{t}, G, H)$ are said to be given with their $(f, \mathbf{x})$*-scaling/displacement generator (G, H) *and* $(\mathbf{s}, \mathbf{t})$-scaling generator (G, H), *both* of length r, *respectively, if*

$$V_f(\mathbf{x}, G, H) = \sum_{k=1}^{r} \text{diag } (f g_i^{(k)} / (f - x_i^n))_{i=0}^{n-1} V(\mathbf{x}) Z_{1/f}^T(\mathbf{h}^{(k)}),$$

$$C(\mathbf{s}, \mathbf{t}, G, H) = \sum_{k=1}^{r} \text{diag } (s_i g_i^{(k)})_{i=0}^{n-1} C(\mathbf{s}, \mathbf{t}) \text{ diag } (h_j^{(k)})_{j=0}^{n-1}$$

where

$$\mathbf{g}^{(k)} = (g_i^{(k)})_{i=0}^{n-1}, \mathbf{h}^{(k)} = (h_j^{(k)})_{j=0}^{n-1}, k = 0, \ldots, n-1, G = (\mathbf{g}^{(k)})_{k=0}^{n-1}, H = (\mathbf{h}^{(k)})_{k=0}^{n-1},$$

$\mathrm{diag}(\mathbf{w}_i)_{i=0}^{n-1}$ *denotes the diagonal matrix with the diagonal entries* $w_0, \ldots, w_{n-1}$; $Z_q(\mathbf{y}) = (y_{i,j})_{i,j=0}^{n-1}$ *is the* q-circulant matrix, $y_{i,j} = y_{i-j \bmod n}$ *for* $i \geq j$, $y_{i,j} = qy_{i-j \bmod n}$ *for* $i < j$, $\mathbf{y} = (y_k)_{k=0}^{n-1}$, $q = 1/f$, *and* f *is a scalar,* $f \neq 0$, $f \neq x_i^n$ *for* $i = 1, \ldots, n-1$. *1-circulant matrices are called* circulant. *The minimum length* r *in the above representations of the matrices* $V_f(\mathbf{x}, G, H)$ *and* $C(\mathbf{s}, \mathbf{t}, G, H)$ *(for all pairs* $(f, \mathbf{x})$ *and* $(\mathbf{s}, \mathbf{t})$*, respectively) is called the* scaling/displacement rank *and the* scaling rank *of these matrices, respectively. If* $r = O(1)$ *as* $n \to \infty$*, then the above matrices are called* Vandermonde-like *and* Cauchy-like matrices, *respectively.*

Note that Vandermonde and Cauchy matrices have scaling/displacement and scaling ranks 1, respectively.

We immediately observe the following results.

Theorem 3.6.2 *In the* n^2*-dimensional linear space of the entries of* $n \times n$ *matrices, the entries of the* $n \times n$ *matrices having scaling/displacement or scaling ranks at most* r *form two algebraic varieties of dimensions at most* $(2r+1)n+1$ *and* $(2r+2)n$*, respectively.*

It is well known that all $k \times k$ Vandermonde and Cauchy matrices $V = (x_i^j)_{i,j=0}^{k-1}$ and $C = (\frac{1}{s_i - t_j})_{i,j=0}^{k-1}$ have generic rank profile (provided that all the values $x_0, \ldots, x_{k-1}$ are distinct and all the values $s_0, t_0, \ldots, s_{k-1}, t_{k-1}$ are distinct). By Theorem 3.6.1, equation (2.2) holds for the matrices $A = VY$ and $A = CY$. By Definition 3.6.1, these are Vandermonde-like and Cauchy-like matrices, respectively, defined with their displacement/scaling generators of length 1 and scaling generators of length 1, respectively. By combining these observations, we obtain the next result.

Theorem 3.6.3 *Equation (2.2) holds generically for (Vandermonde-like matrices) having displacement/scaling rank at most* r *and for (Cauchy-like) matrices having scaling rank at most* r*, for any* $r \geq 1$.

We also have the next trivial corollary of Theorem 3.6.1.

Corollary 3.6.1 *Equation (2.2) holds generically for $n \times n$ sparse matrices with a fixed pattern of their nonzero entries as long as this pattern allows a matrix to have generic rank profile.*

3.7 Toeplitz and Toeplitz-like Matrices: Definitions, a Basic Property and Equation (3.2.2)

Our study of Toeplitz, Toeplitz-like, Hankel, and Hankel-like matrices throughout section 3.9 will be valid over any field $\mathbf{F}$, though for our study of the eigenproblem we only need the special case where $\mathbf{F} = \mathbf{C}$ and $\mathbf{C}$ denotes the field of complex numbers. Frobenius matrices are not involved in this and the next sections, and we will use capital F to denote the linear operators of shift (displacement).

Definition 3.7.1 *A matrix $T = (t_{i,j})$ is called a* Toeplitz matrix *if*

$$t_{i+1,j+1} = t_{i,j} \tag{7.1}$$

for all pairs i, j for which $t_{i,j}$ and $t_{i+1,j+1}$ are defined.

Toeplitz matrices are a special class (where $r \leq 2$) of the following important and well-studied class of Toeplitz-like matrices having a small displacement rank r (cf. [BP94], pp. 174-211).

Definition 3.7.2 *(cf. e.g. [BP94], Definition 11.1). For an $n \times n$ matrix T, define the two displacement generators,*

$$F_-(T) = T - Z^T T Z, \quad F_+(T) = T - Z T Z^T, \tag{7.2}$$

where

$$Z = \begin{pmatrix} 0 & & & 0 \\ 1 & 0 & & \\ & 1 & \ddots & \\ & & \ddots & \\ 0 & & 1 & 0 \end{pmatrix}$$

is a down-shift $n \times n$ matrix. If for $F = F_+$ or $F = F_-$ we have

$$F(T) = GH^T, \tag{7.3}$$

where G and H are $n \times r$ matrices, then the pair of matrices (G, H) is called an F- generator *or a* displacement generator *of T of length r and will be denoted d.g.$_r(T)$. The minimum r allowing the above representation (7.3) is called the F-*rank *or* displacement rank *of T. T is called a* Toeplitz-like matrix *if $r = O(1)$ as $n \to \infty$.*

Next, we will recall a basic property of Toeplitz-like matrices, which will enable us to manipulate with them by means of operating with a few entries of their short displacement generators, rather than with their own more numerous entries.

Theorem 3.7.1 *[KKM79]. Let F_-, F_+, T, G, H, and r be as in (7.2) and (7.3). Then $F(T) = GH^T = \sum_{k=1}^{r} \mathbf{g}^{(k)}\mathbf{h}^{(k)T}$ if and only if*

$$T = \sum_{k=1}^{r} L^T(\mathbf{g}^{(k)})L(\mathbf{h}^{(k)}) \text{ for } F = F_-, \quad T = \sum_{k=1}^{r} L(\mathbf{g}^{(k)})L^T(\mathbf{h}^{(k)}) \text{ for } F = F_+, \tag{7.4}$$

where $G = (\mathbf{g}^{(1)}, \dots, \mathbf{g}^{(r)})$, $H = (\mathbf{h}^{(1)}, \dots, \mathbf{h}^{(r)})$, and $L(\mathbf{v})$ is a lower triangular Toeplitz matrix with the first column $\mathbf{v}$.

Based on equations (7.1)-(7.4), we immediately obtain the following results.

Theorem 3.7.2 *The class of all $n \times n$ Toeplitz matrices forms a linear subspace T of dimension $2n - 1$ in the space of $n \times n$ matrices.*

Theorem 3.7.3 *The class of all $n \times n$ (Toeplitz-like) matrices having displacement rank at most r forms an algebraic variety T_r of dimension at most $2rn$ in the space of $n \times n$ matrices. Furthermore, $T \subset T_r$ for $r \geq 2$.*

Now we immediately verify that (2.2) holds for the Toeplitz matrix $A = Z$. Combining the latter observation with Theorems 3.7.2, 3.7.3 and Corollary 3.5.1 gives us the next result.

Corollary 3.7.1 *Equation (2.2) holds generically for $n \times n$ lower (upper) triangular Toeplitz matrices and $n \times n$ Toeplitz matrices as well as for $n \times n$ (Toeplitz-like) matrices of displacement rank at most r for any $r \geq 1$.*

Theorem 3.7.4 *Equation (2.2) holds generically for $n \times n$ circulant matrices A (which are a special case of Toeplitz matrices, cf. Definition 3.6.2).*

Proof. Recall a simple explicit expression of the eigenvalues of a circulant matrix via the entries (cf. e.g. [BP94], Theorem 5.1 on p.134). Based on this expression, define a circulant matrix having n distinct eigenvalues and recall Remark 3.5.1. $\square$

3.8 Hankel, Hankel-like, and Hankel-like+Toeplitz-like Matrices: Definitions and Equation (3.2.2)

Hankel matrices and *Hankel-like matrices* of displacement rank r are obtained from Toeplitz matrices and Toeplitz-like matrices of displacement rank r, respectively, by their pre-multiplication (or post-multiplication) by the *reflection matrix* J having ones on its antidiagonal and zero entries elsewhere. (Note that J^2 is the identity matrix.)

By Theorems 3.7.2 and 3.7.3, *the entries of $n \times n$ Hankel matrices $H = (h_{i+j})_{i,j=0}^{k-1}$ form a linear space of dimension $2n - 1$, whereas Hankel-like matrices of displacement rank at most r form an algebraic variety of dimension at most $2rn$ in $\mathbf{F}^{n \times n}$.*

Now we recall the following result from [KS91].

Theorem 3.8.1 *Let W be an $n \times n$ nonsingular matrix. Let U^T and L be generic lower triangular $n \times n$ Toeplitz matrices, and let U have its diagonal entries equal to 1. Then the matrix $A = UWL$ satisfies equation (2.2).*

By applying Theorem 3.8.1 to the (triangular) Hankel matrix $W = J(I + Z)$, we obtain that (2.2) holds for the triangular Hankel matrix $A = UJ(I + Z)L$. We note that A has displacement rank 1 because JA is a lower triangular Toeplitz matrix.

Corollary 3.8.1 *Equation (2.2) holds generically for $n \times n$ lower (upper) triangular Hankel matrices and for $n \times n$ Hankel matrices as well as for $n \times n$ (Hankel-like) matrices of displacement rank at most r for any $r \geq 1$.*

Clearly, Corollary 3.8.1 can be immediately extended to the class of Toeplitz-like +Hankel-like matrices.

3.9 Fast Randomized Solution of a Consistent Toeplitz-like Linear System of Equations over an Arbitrary Field

In this section we will follow the appendix of the paper [Kal95] and slightly strengthen its results (cf. Remark 3.9.3). We will first recall the algorithm that supports the next theorem and then will extend this algorithm to complete the proof of Theorem 3.5.4.

Theorem 3.9.1 *Let A be an $n \times n$ matrix having generic rank profile (cf. Definition 3.6.1) and represented as*

$$A = \sum_{k=1}^{r} L^T(\mathbf{g}^{(k)})L(\mathbf{h}^{(k)}). \tag{9.1}$$

*Then $O(n \log^2 n)$ ops suffice to compute $\rho = \text{rank}(A)$ and $2r$ vectors $\mathbf{u}^{(1)}, \mathbf{v}^{(1)}, \ldots, \mathbf{u}^{(r)}, \mathbf{v}^{(r)}$ of dimension n defining an F_+-*generator *of A_ρ^{-1}, that is, satisfying the matrix equation $A_\rho^{-1} = \sum_{k=1}^{r} L(\mathbf{u}^{(k)})L^T(\mathbf{v}^{(k)})$, where A_ρ is the $\rho \times \rho$ northwestern (leading principal) submatrix of A, $A_\rho = A^{-1}$ if A is (strongly) nonsingular.*

Remark 3.9.1 *For the cost estimates of Theorem 3.9.1, we assumed computations over the complex field, but the algorithms supporting the theorem can be applied over any field of constants with the standard minor adjustment of the cost bounds.*

We will start with some definitions and auxiliary results.

Definition 3.9.1 *(cf. [Str69], [BP94], p. 99). Let*

$$A = \begin{pmatrix} B & C \\ E & K \end{pmatrix}$$

be a nonsingular $n \times n$ matrix, with the nonsingular $k \times k$ submatrix B, $k = \lceil \frac{n}{2} \rceil$. Then the balanced triangular factorization (BTF) *of A and A^{-1} is given by the next matrix equations:*

$$A = \begin{pmatrix} I_k & O \\ EB^{-1} & I_{n-k} \end{pmatrix} \begin{pmatrix} B & O \\ O & S \end{pmatrix} \begin{pmatrix} I_k & B^{-1}C \\ O & I_{n-k} \end{pmatrix}, \tag{9.2}$$

$$A^{-1} = \begin{pmatrix} I_k & -B^{-1}C \\ O & I_{n-k} \end{pmatrix} \begin{pmatrix} B^{-1} & O \\ O & S^{-1} \end{pmatrix} \begin{pmatrix} I_k & O \\ -EB^{-1} & I_{n-k} \end{pmatrix}, \tag{9.3}$$

where

$$S = K - EB^{-1}C \tag{9.4}$$

is the Schur complement of B *in* A. *The BTF of* A *and* A^{-1} *can be extended to* B, S, B^{-1} *and* S^{-1} *and recursively to their submatrices and the Schur complements as long as we deal with nonempty nonsingular matrices. If nonsingularity holds throughout all recursive steps, then we arrive at the* BRTF *(*balanced recursive triangular factorization*) of* A *and* A^{-1}.

The BRTF of a matrix can be obtained by the Gauss-Jordan elimination algorithm, and we have the following simple results.

Theorem 3.9.2 *If* A *and* B *are nonsingular, then* S^{-1} *is the southeastern (trailing principal) submatrix of* A^{-1}.

Proof. Theorem immediately follows from (9.3). □

Theorem 3.9.3 *(cf. [BP94]), exercise 4 on p.212). A matrix* A *has BRTF if it is strongly nonsingular (cf. Definition 3.6.1).*

For a matrix A having generic rank profile, the BRTF has a natural extension, which can be efficiently computed by a divide-and-conquer algorithm We are going to describe this algorithm assuming that A is a Toeplitz-like matrix. We will start with some auxiliary results.

Theorem 3.9.4 *[BA80]. For any* $n \times n$ *matrix* A,

$$\text{rank}(F_-(A)) - 2 \leq \text{rank}(F_+(A)) \leq \text{rank} F_-(A) + 2.$$

Furthermore, given a $d.g._r(T)$ *under* $F = F_+$ *(resp.* $F = F_-$ *), it suffices to use* $O(r)$ *ops in order to compute a* $d.g._{r+2}(T)$ *under* $F = F_-$ *(resp.* $F = F_+$ *).*

Theorem 3.9.5 (cf. e.g. [BP94], Corollary 12.1 *). Let* T_1 *and* T_2 *be two Toeplitz-like matrices, given with their* F*-generators of lengths* r_1 *and* r_2*, respectively, for* $F = F_+$ *or* $F = F_-$*. Then an* F*-generator of length at most* $r_1 + r_2 + 1$ *for the matrix* T_1T_2 *can be computed by using* $O(r_1r_2)$ *multiplications of polynomials of degree* $O(n)$ *and* $O(r_1 + r_2)$ *summations of* $O(r_1 + r_2)$ *vectors of dimension* n*, at the overall cost of* *performing $O((r_1 + r_2)^2 n \log n)$ ops. Furthermore, a d.g.$_r(UAL)$ for a given d.g.$_r(A)$ and a given pair of lower triangular Toeplitz matrices L and U^T can be computed at the cost $O(r^2 n \log n)$, provided that $F = F_-$.*

Theorem 3.9.6 *(cf. Proposition A.6 of [P92], [P93], or [BP94], Problem 2.2.11b). Given an d.g.$_{\hat{r}}(A) = (G, H)$ and the displacement rank r of A, $r < \hat{r} \leq n$, one can compute d.g.$_r(A)$ by using $O(\hat{r}^2 n)$ ops.*

Theorem 3.9.7 *[KKM79]. Let A be a nonsingular matrix. Then we have* rank $(F_+(A^{-1})) = \text{rank}\ (F_-(A))$.

Theorem 3.9.8 *(cf. [M80], [BA80], [BP94]). Let A be an $n \times n$ strongly nonsingular Toeplitz-like matrix such that*

$$A = \begin{pmatrix} B & C \\ E & K \end{pmatrix}, \qquad S = K - EB^{-1}C,$$

B is a $k \times k$ matrix, and S is the $(n-k) \times (n-k)$ Schur complement of B in A. Let $r = \text{rank}(F_+(A))$. *Then*

$$\text{rank}\ (F_-(S^{-1})) = \text{rank}\ (F_+(S)) \leq r,$$

$$\text{rank}\ (F_-(B^{-1})) = \text{rank}\ (F_+(B)) \leq r,$$

$$\text{rank}\ (F_+(S^{-1})) = \text{rank}\ (F_-(S)) \leq r + 2,$$

$$\text{rank}\ (F_+(B^{-1})) = \text{rank}\ (F_-(B)) \leq r + 2.$$

Proof. Theorem 3.9.6 follows from Theorems 3.9.2, 3.9.4 and 3.9.7. □

(We slightly abuse the notation when we write S, but the Schur complement S and a finite set S will never appear together in this paper and thus will not interfere with each other.)

Given $d.g_{r}.(A)$ and two lower triangular Toeplitz matrices U^T and L, we may compute $d.g._r(\tilde{A})$ for $\tilde{A} = UAL$ and $F = F_-$ at the cost of performing at most $O(r^2 n \log n)$ ops due to Theorem 3.9.5.

Now, we will describe an algorithm supporting Theorem 3.9.1.

Algorithm 3.9.1 *computation of the largest northwestern (leading principal) inverse.*

Input: n-dimensional vectors $\mathbf{g}^{(1)}, \mathbf{h}^{(1)}, \ldots, \mathbf{g}^{(r)}, \mathbf{h}^{(r)}$ such that a Toeplitz-like matrix

$$A = \sum_{i=1}^{r} L^T(\mathbf{g}^{(i)})L(\mathbf{h}^{(i)})$$

has generic rank profile.

Output: An integer $\rho \leq n$ and n-dimensional vectors $\mathbf{u}^{(1)}, \ \mathbf{v}^{(1)}, \ldots, \mathbf{u}^{(r)}, \ \mathbf{v}^{(r)}$, such that $\rho = \text{rank}(A)$ and

$$A_\rho^{-1} = \sum_{m=1}^{r} L(\mathbf{u}^{(m)})L^T(\mathbf{v}^{(m)}),$$

where A_ρ denote the $\rho \times \rho$ northwestern submatrix of A.

1. Represent A as a 2×2 block matrix $\begin{pmatrix} B & C \\ E & K \end{pmatrix}$, for $k = \lceil \frac{n}{2} \rceil$, where the $k \times k$ submatrix B of A is singular if and only if $k > \rho$ (since A has generic rank profile). Apply Algorithm 3.9.1 recursively to the input matrix B replacing A. (Note that the first k components of all the given vectors $\mathbf{g}^{(i)}$ and $\mathbf{h}^{(i)}$ (for all i) define a $d.g._r(B)$.) If $\rho \geq k$, the output of this stage is the desired output of the algorithm. Otherwise, the matrix B is nonsingular, and then we obtain a $d.g._{r+2}(B^{-1})$ for $F = F_-$ and a $d.g._r(B^{-1})$ for $F = F_+$.

2. Apply Theorem 3.9.5 to compute a $d.g._r(S)$ for the matrix $S = K - E\ B^{-1}C$ and for $F = F_+$.

3. Apply the algorithm recursively to the Toeplitz-like input matrix S, replacing A. Output $\rho = \text{rank}(A) = k + \text{rank}(S)$.

4. By using Theorems 3.7.1, 3.9.4-3.9.8, compute $s.g._r(A_\rho^{-1})$ for $F = F_+$ (see some further comments below).

Let us specify stage 4. Consider A_ρ, the $\rho \times \rho$ northwestern (leading principal) submatrix of A, $A_\rho = \begin{pmatrix} B & V \\ D & R \end{pmatrix}$. Write $\breve{S} = R - D\ B^{-1}V$. Note that at the preceding stages we have computed $d.g._r(V)$ and $d.g._r(D)$ for $F = F_-$, $d.g._r(B^{-1})$, $d.g._{2r+1}(-B^{-1}V)$, $d.g._{2r+1}(-DB^{-1})$, and $d.g._r(\breve{S}^{-1})$ for $F = F_+$. We obtain the following block representation:

$$A_\rho^{-1} = \begin{pmatrix} M_{1,1} & M_{1,2} \\ M_{2,1} & \breve{S}^{-1} \end{pmatrix},$$

where $M_{1,2} = -B^{-1}V\ \breve{S}^{-1}$, $M_{2,1} = -\breve{S}^{-1}D\ B^{-1}$, $M_{1,1} = B^{-1} - M_{1,2}DB^{-1}$. By applying Theorems 3.7.1, 3.9.4-3.9.8 we compute $d.g._r(A_\rho^{-1})$ for $F = F_+$. □

Combining (2.7) with Algorithm 3.9.1 and Theorems 3.7.1, 3.9.2-3.9.8, we immediately obtain Theorem 3.9.1. □

Furthermore, Algorithm 3.9.1 is immediately extended to the solution of a consistent linear system $A\mathbf{x} = \mathbf{b}$, whereas the assumption that the matrix A has generic rank profile is relaxed due to the following result of [KS91].

Theorem 3.9.9 *Let S be a fixed finite set. Let $L(\mathbf{g})$ and $L(\mathbf{h})$ be $n \times n$ unit lower triangular Toeplitz matrices, each defined by the $n-1$ random entries of its first column, which are uniformly sampled from the set S. Let A be an $n \times n$ matrix of rank ρ. Then the matrix $L^T(\mathbf{g})AL(\mathbf{h})$ has generic rank profile with a probability at least $1-(\rho+1)\rho/(2|S|)$.*

Due to the latter theorem, we may replace the matrix A by the matrix $L^T(\mathbf{g})AL(\mathbf{h})$ and the linear system $A\mathbf{x} = \mathbf{b}$ by the linear system $L^T(\mathbf{g})AL(\mathbf{h})\mathbf{y} = L^T(\mathbf{g})\mathbf{b}$ where $\mathbf{x} = L(\mathbf{h})\mathbf{y}$. Furthermore, (9.1) turns into the equation $L^T(\mathbf{g})AL(\mathbf{h}) = \sum_{k=1}^{r} L^T(\tilde{\mathbf{g}}^{(k)})L^T(\tilde{\mathbf{h}}^{(k)})$, where $\tilde{\mathbf{g}}^{(k)}$ and $\tilde{\mathbf{h}}^{(k)}$ are the convolutions of the pairs of vectors $(\mathbf{g}, \mathbf{g}^{(k)})$ and $(\mathbf{h}, \mathbf{h}^{(k)})$, respectively.

The extension from Toeplitz-like representation (9.1) to Hankel-like with the same generator length is immediate (by means of the pre- or post-multiplication of equation (9.1) by the reflection matrix J).

These observations imply the following result.

Theorem 3.9.10 *To compute the rank ρ of an $n \times n$ matrix A represented in the form (9.1) and to solve a consistent linear system of equations $A\mathbf{x} = \mathbf{b}$ with such a matrix of coefficients, it suffices to sample uniformly $2n-2$ random values from a fixed finite subset S and to perform $O(r^2 n \log^2 n)$ ops. The resulting algorithm may fail with a probability at most $(\rho+1)\rho/(2|S|) \leq (n+1)n/(2|S|)$ but otherwise produces correct output. The same results apply to the computations with the matrices AJ and JA replacing A, where J is the reflection matrix, having ones on its antidiagonal and zero entries elsewhere.*

Now the proof of Theorem 3.5.4 can be immediately completed due to Theorem 3.9.1 applied to AJ for a Toeplitz matrix A. □

Remark 3.9.2 *The proof of Theorem 3.9.8 applies to Toeplitz-like and to Hankel-like matrices A but can be easily extended to Toeplitz-like + Hankel-like matrices and some* *other classes of structured matrices by using their known representations via the associated linear operators (cf. [BP94]).*

Remark 3.9.3 *Algorithm 3.9.1 and the proof of Theorem 3.9.1 essentially follow the line of the appendix of [Kal95], which extends the MBA algorithm of [M80], [BA80], to the case of singular input. Due to Theorem 3.9.6, we derandomize Theorem 3.9.1, thus decreasing the number of random parameters and the failure probability estimate in Theorem3.9.10. The computation at stages 1 and 2 of Algorithm 3.9.1 can be a little simplified further (cf. Lemma 3.1 of [OP98]).*

3.10 Randomized Reduction of a Matrix to the Tridiagonal Form

A popular alternative to the Frobenius reduction is the reduction to the tridiagonal form, to which a generic $n \times n$ matrix A can be reduced by means of an unsymmetric variation of Lanczos randomized algorithm (cf. [Par80], [BP94], pp. 122-123, 172, 325). To facilitate the analysis of this algorithm, we will next present it in the form based on factorization of the inverse of a Hankel matrix $H^{(0)}$ (which is a *Bezout matrix,* cf. [BP94], p. 160) rather than on the more customary computation of three term recurrence relations (cf. [GL96], pp. 503-504).

Algorithm 3.10.1 *randomized tridiagonalization of a generic matrix by a modified Lanczos algorithm.*

Input: an $n \times n$ matrix A.
Output: FAILURE or a triple of $n \times n$ matrices P, Q and $\bar{T}$ such that

$$P^H Q = D, \ \bar{T} = P^H A Q, \tag{10.1}$$

D is a diagonal matrix and T is a complex symmetric tridiagonal matrix.

Computations:

1. Fix a pair of n-dimensional vectors $\mathbf{u}$ and $\mathbf{v}$ and compute and output the Krylov matrices $U = K(A^H, \mathbf{u}, n)$ and $V = K(A, \mathbf{v}, n)$.
2. Compute the Hankel matrix $H^{(0)} = (\mathbf{u}^H A^{i+j} \mathbf{v})_{i,j=0}^{n-1}$.

3. Apply the MBA algorithm of [M80], [BA80], that is, Algorithm 3.8.1 trivially modified so that it either computes the balanced recursive triangular factorization (BRTF) of the matrices $H^{(0)}$ and $(H^{(0)})^{-1}$ whenever the matrix $H^{(0)}$ is nonsingular and has BRTF or outputs FAILURE otherwise.

4. Compress the BRTF of $(H^{(0)})^{-1}$ to compute the unit lower triangular matrix L and the diagonal matrix D satisfying $(H^{(0)})^{-1} = L^T DL$.

5. Compute and output the matrices $P = UL^T$, $Q = VL^T$ and $\tilde{T} = P^H AQ$.

Now, to yield formal tridiagonal reduction of A, it suffices to compute the matrices $W = QD^{-1}$ and $T = \tilde{T}D^{-1}$, which, by (10.1), satisfy

$$T = W^{-1}AW. \tag{10.2}$$

Correctness of Algorithm 3.9.1 is proved in [Par80], [BP94]. The algorithm outputs FAILURE if and only if the matrix $H^{(0)}$ is singular or has no BRTF.

How likely is the failure of the algorithm ? By Theorem 3.9.3, Algorithm 3.9.1 does not fail if the matrix $H^{(0)}$ is strongly nonsingular, that is, unless at least one of its $k \times k$ northwestern submatrices $H_k^{(0)}$, $k = 1, \ldots, n$, is singular. The singularity of $H_k^{(0)}$ imposes the polynomial equation, $\det H_k^{(0)} = 0$, on the entries of $\mathbf{u}, \mathbf{v}$ and A. Such an equation has degree k in the entries of each of the vectors $\mathbf{u}$ and $\mathbf{v}$. For a fixed pair of nonzero vectors $\mathbf{u}$ and $\mathbf{v}$ and for generic A, such an equation holds for no k, and, therefore, the algorithm does not fail. For generic A and random $\mathbf{u}$ and $\mathbf{v}$, we apply Theorem 3.5.3. Summarizing this analysis, we obtain the next results.

Theorem 3.10.1 *Algorithm 3.9.1 does not fail if* $\mathbf{u}$ *and* $\mathbf{v}$ *are two nonzero vectors and if A is a generic $n \times n$ matrix. If, for a fixed $n \times n$ matrix A and for some pair of vectors* $\mathbf{u}$ *and* $\mathbf{v}$*, the matrix $H^{(0)}$ is strongly nonsingular, then for such a matrix A and for the vectors* $\mathbf{u}$ *and* $\mathbf{v}$ *with entries uniformly sampled from a fixed finite set S, Algorithm 3.9.1 may fail with a probability at most* $(n+1)n/(2|S|)$.

Remark 3.10.1 *If A is a nonsingular real symmetric or Hermitian matrix, then $P = Q$, $D = I$, and Algorithm 3.9.1 never fails for generic vectors* $\mathbf{u}$ *and* $\mathbf{v}$*. (To yield nonsingularity, one may shift to the family of matrices $A - bI_n$ for a scalar b and observe that*

$Q^H AQ = T$ if and only if $Q^H(A - bI_n)Q = T - bI_n$.) In this case, the resulting tridiagonal matrix T is real symmetric and the subsequent approximation of the eigenvalues is considerably simplifed [BP91], [BP98].

We also have the following result.

Theorem 3.10.2 *The cost of performing Stages 1-4 of Algorithm 3.9.1 is bounded by $O(K_A(n) + K_{A^H}(n) + n^2 \log n)$ ops, which turns into $O(n^2 \log n)$ for Toeplitz-like and Hankel-like matrices A and into $O(n^2 \log^2 n)$ for Vandermonde-like and Cauchy-like matrices A.*

Proof: Clearly, the claimed cost bound applies at stages 1 and 2. Stage 3 requires $O(n \log^2 n)$ ops (see section 3.9), and this bound is immediately extended to the computation of the BRTF of $(H^{(0)})^{-1}$ at the same cost. Since all matrices defining the BRTF are Hankel-like or Toeplitz-like matrices, $O(n^2 \log n)$ ops suffice for the compression of the BRTF at stage 4 (which gives us the matrices L and D). □

Remark 3.10.2 *The term $K_{A^H}(0)$ can be deleted from the above cost bound due to Tellegen's theorem [PSD70].*

Stage 5 of Algorithm 3.9.1 requires order of $M(n)$ ops (note that the structure of A is not preserved in the transition to the matrices L^T, U, and V), but next we will show how to stay within the superior cost bound of Theorem 3.10.2 if we agree to represent the tridiagonalizaiton of A by the matrices V, D, L^T and T such that the matrices T and $W = VL^T D^{-1}$ satisfy (10.2).

Algorithm 3.10.2 *accelerated randomized tridiagonalization.*

Input: an $n \times n$ matrix A.

Output: FAILURE or four matrices D (diagonal), L (lower triangular), T (complex tridiagonal) and V such that the matrix $W = VL^T D^{-1}$ is nonsingular and satisfies (10.2).

Computations:

1.–4. as in Algorithm 3.10.1, except that the computation of the Krylov matrix U at stage 1 can be omitted.

5. Compute the matrix $H^{(1)} = (\mathbf{u}^H A^{i+j}\mathbf{v})_{i,j=1}^{n}$ (most of the entries of $H^{(1)}$ are also the entries of $H^{(0)}$).

6. Compute and output the $n \times n$ tridiagonal matrix T formed by the three main diagonals of the matrix $LH^{(1)}L^T D^{-1}$.

The correctness proof for this algorithm can be found in [Par80].

Since $H^{(1)}$ is a Hankel matrix, the computation of the matrix $B = H^{(1)}L^T D^{-1}$ requires $O(n^2 \log n)$ ops (cf. (2.7)). Clearly, $O(n^2)$ ops suffice to compute the three main diagonals of the matrix LB.

Summarizing our analysis of Algorithm 3.10.1 and 3.10.2, we obtain the next result.

Theorem 3.10.3 *Let Algorithm 3.10.2 be applied to generic $n \times n$ matrix A and let the $2n$ coordinates of the vectors $\mathbf{u}$ and $\mathbf{v}$ be uniformly sampled from a fixed finite set S. Then the algorithm either outputs FAILURE with a probability of most $(n+1)n/(2|S|)$ or computes an $n \times n$ tridiagonal matrix T and three nonsingular $n \times n$ matrices, V, L (tower triangular) and D(diagonal), such that the matrices T and $W = VL^T D^{-1}$ satisfy (10.2). Not counting the cost of the random sampling, the entire computation by Algorithm 3.10.2 requires $O(K_A(n) + n^2 \log n)$ ops.*

By Remark 3.3.4, the characteristic polynomial $c_A(x)$ is immediately available as soon as the matrix A has been reduced to the Frobenius form or to the triangular Frobenius form. For the tridiagonal reduction, the situation is similar:

Theorem 3.10.4 *(cf. e.g. [BP91], [BP98]). $O(n)$ ops suffice to obtain the characteristic polynomial of an $n \times n$ tridiagonal matrix.*

3.11 Approximation of the Eigenvalues and the Computation of Their Algebraic Multiplicities

By Theorem 3.3.1, every eigenvalue of A is a zero of its both minimum and characteristic polynomials. Suppose that we already know such a polynomial (cf. Theorems 3.4.1, 3.5.1, 3.5.2, 3.5.4 and 3.5.5). In this case we may first easily estimate from above the absolute values of all the eigenvalues by applying either Gershgorin's theorem to A (cf. [GL96]) or the known root radius estimates for polynomials (cf. [P97]), then scale A to move all the eigenvalues into the unit disc $\{x : |x| \leq 1\}$ (cf. Remark 3.3.1), and finally approximate them by applying the polynomial rootfinders of [P95b], [P96] to the polynomials $c_A(x)$ or $m_A(x)$ (see [P97] on some alternative effective polynomial rootfinders). This gives us the following result.

Theorem 3.11.1 *All the eigenvalues of an $n \times n$ matrix A given with its minimum or characteristic polynomials can be approximated within the error bound $2^{-b}||A||$, at the cost of performing $O((n \log^2 n)(\log b + \log^2 n))$ ops.*

The algorithms of [P95b], [P96] do not compute polynomial zeros exactly, which makes their application to $c_A(x)$ inefficient for the task of computing the algebraic multiplicities μ_j^+ of the eigenvalues λ_j. We may, however, circumvent this obstacle easily by exploiting the following simple result.

Theorem 3.11.2 *A zero $x = z$ of a polynomial $p(x)$ has multiplicity $m \geq k \geq 0$ if and only if it is a zero of $p^{(k)}(x)$ of multiplicity $m - k$.*

Definition 3.11.1 *Let gcd $(u_0(x)), \ldots, u_m(x))$ denote the greatest common divisor of polynomials $u_0(x), \ldots, u_m(x)$. For a fixed polynomial $p(x)$ with coefficient vector $\mathbf{p}$, let us write $d_{\mathbf{p},0}(x) = p(x)$, $d_{\mathbf{p},m}(x) = \gcd(p^{(0)}(x), \ldots, p^{(m)}(x))$, $f_{\mathbf{p},m}(x) = d_{\mathbf{p},m-1}(x)/d_{\mathbf{p},m}(x)$, $m = 1, \ldots, n$.*

We have the following corollary of Theorem 3.11.2.

Corollary 3.11.1 *$x = z$ is a zero of $p(x)$ having multiplicity $m \geq 0$ if and only if it is a zero of $f_{\mathbf{p},m}(x)$.*

Corollary 3.11.1 suggests the following algorithm (cf. [Y76]).

Algorithm 3.11.1 *approximation of the zeros of a polynomial and computation of their multiplicities.*

Input: the coefficient vector $\mathbf{p}$ of a polynomial $p(x)$ of degree n having all its zeros $z_1, \ldots, z_n$ in the unit disc $\{x : |x| \leq 1\}$, and $\epsilon = 2^{-b}, b > 0$.
Output: k pairs (z_j^*, μ_j^+), $j = 1, \ldots, k$, where $|z_j^* - z_j| \leq \epsilon$, z_j is a zero of $p(x)$ of multiplicity μ_j^+, $j = 1, \ldots, k$,

Computations.

1. Compute $d_{\mathbf{p},m}(x)$ for $m = 1, \ldots, n$.

2. Compute $f_{\mathbf{p},m}(x)$ for $m = 1, \ldots, n$.

3. Approximate within ϵ all the roots of $f_{\mathbf{p},m}(x)$ for $m = 1, \ldots, n$. For each m, $m = 1, \ldots, n$, output the computed approximations $z^*_{j(m)}$ to the roots $z_{j(m)}$ of $f_{\mathbf{p},m}(x)$, together with their multiplicity $m = \mu^+_{j(m)}$.

Correctness of the algorithm follows from Corollary 3.11.1. Stages 1 and 2 can be performed by recursive application of Euclidean algorithm and straightforward polynomial division, respectively, at the overall cost of performing $O(n^2)$ ops. At stage 3 we apply the algorithm of [P95b], [P96], which requires $O((n \log n)(\log b + \log^2 n))$ ops.

Summarizing, we perform Algorithm 3.11.1 by using $O(n^2 + (n \log n) \log b)$ ops.

Due to Theorem 3.3.1, Algorithm 3.11.1 applied to the polynomial $p(x) = c_A(x)$ outputs approximations to the eigenvalues λ_j of A and their algebraic multiplicities μ^+_j.

3.12 Computation of the Eigenspaces of Given Eigenvalues and the Proof of the Main Theorems

By Definition 3.3.1, the eigenspace of an eigenvalue λ_j of A is the null space of the matrix

$$A_j = \lambda_j I_n - A, \tag{12.1}$$

having dimension

$$\mu_j = n - \text{rank } A_j. \tag{12.2}$$

In this section we will study the following eigenspace problem.

Problem 3.12.1 *the eigenspace computation. Given an $n \times n$ matrix A and all its eigenvalues (without their algebraic and geometric multiplicities), compute some bases for the eigenspaces of all these eigenvalues.*

The solution of 3.12.1 depends on the form in which the matrix A is given. We will prove the following results (cf. Definitions 3.3.6 and 3.3.7).

Theorem 3.12.1 *$O(n^3)$ ops suffice to solve 3.12.1 if its input matrix is in the triangular Frobenius form or (more generally) is a Hessenberg matrix.*

Theorem 3.12.2 *$O(n^2)$ ops suffice to solve 3.12.1 if its input matrix is tridiagonal or is in the diagonal Frobenius form.*

The proofs will rely on the next two theorems.

Theorem 3.12.3 *Let B be a nonsingular $(n-\mu)\times(n-\mu)$ northwestern submatrix of $n\times n$ matrix $W=\begin{pmatrix} B & C \\ E & G \end{pmatrix}$ of rank r, where $n\geq r\geq n-\mu$. Let S denote the Schur complement of B in W, that is, let $S=G-EB^{-1}C$. Let the columns of a $\mu\times(n-r)$ matrix M form a basis of the null space of S. Then the $n-r$ columns of the $n\times(n-r)$ matrix*

$$N=\begin{pmatrix} I_{n-\mu} & -B^{-1}C \\ 0 & I_\mu \end{pmatrix}\begin{pmatrix} 0 \\ M \end{pmatrix} \qquad (12.3)$$

form a basis of the null space of W.

Proof. Observe that

$$W=\begin{pmatrix} I_{n-\mu} & 0 \\ EB^{-1} & I_\mu \end{pmatrix}\begin{pmatrix} B & 0 \\ 0 & S \end{pmatrix}\begin{pmatrix} I_{n-\mu} & B^{-1}C \\ 0 & I_\mu \end{pmatrix}, \qquad (12.4)$$

$$\begin{pmatrix} I_{n-\mu} & -B^{-1}C \\ 0 & I_\mu \end{pmatrix}=\begin{pmatrix} I_{n-\mu} & B^{-1}C \\ 0 & I_\mu \end{pmatrix}^{-1}.$$

Multiply W by N and substitute (12.3) and (12.4) into the product, to turn it into a null matrix. This shows that all the $n-r$ linearly independent columns of N are in the null space of W, whereas the dimension of this null space is exactly $n-r$. □

Theorem 3.12.4 *Let an $n\times n$ Hessenberg matrix A have exactly k unreduced Hessenberg diagonal blocks H_s of sizes $\rho_s\times\rho_s$, $s=1,\ldots,k$. Let λ_j be an eigenvalue of A having algebraic multiplicity μ_j^+ and geometric multiplicity μ_j. Let exactly μ unreduced diagonal blocks $\lambda_j I_{\rho_s}-H_s$ of the matrix A_j of (12.1) be singular. For every such a singular block, let the row and the column of A_j containing the first row and the last column of the block, respectively, be deleted. Let B denote the resulting submatrix of A_j. Then*

a) B is a nonsingular $(n-\mu)\times(n-\mu)$ matrix,

b) $\mu_j\leq\mu\leq\{\min k,\mu_j^+\}$,

c) $n-\mu=$ rank A_j if A is a tridiagonal matrix or is block diagonal with unreduced Hessenberg diagonal blocks, *d) singularity or nonsingularity of $\lambda_j I_{\rho_s} - H_s$, the s-th diagonal block of A_j, can be decided in $O(\rho_s^2)$ ops; this bound turns into $O(\sum_{s=1}^{k} \rho_s^2) = O(n^2)$ ops for deciding the singularity or nonsingularity of every one of the k blocks,*

e) the latter bounds turn into $O(\rho_s)$ and $O(\sum_{s=1}^{k} \rho_s) = O(n)$, respectively, if the matrix H is tridiagonal or is in the triangular Frobenius form.

Proof. a) Observe that B is an $(n-\mu) \times (n-\mu)$ block triangular matrix. Its diagonal blocks are either nonsingular blocks $\lambda_j I_{\rho_s} - H_s$ or their nonsingular triangular submatrices.

b) The inequality $\mu_j \leq \mu$ follows from part a) and from (12.2). Clearly, $\mu \leq k$. The bound $\mu \leq \mu_j^+$ follows because $c_A(x) = \prod_{s=1}^{k} c_{H_s}(x)$.

c) If A is a tridiagonal matrix or a block diagonal matrix with unreduced $\rho_s \times \rho_s$ Hessenberg blocks H_s, $s = 1, \ldots, k$, then for each s, the deleted column of A_j containing the last column of H_s is a linear combination of the $\rho_s - 1$ preceding columns of A_j, and a similar property holds for the deleted rows of A_j. Therefore, in this case, B is a nonsingular submatrix of A_j having the maximum size, that is, $n - \mu = \text{rank } A_j$.

Finally, d) and e) follow from Theorem 3.3.4. □

Proof of Theorem 3.12.2 By combining Theorems 3.12.3 and 3.12.4 a), c) and e), we compute a basis of the eigenspace of λ_j for any fixed j by using $O((n-\mu)\mu) = O(n\mu)$ ops provided that A is a tridiagonal matrix or is in the diagonal Frobenius form. By Theorem 3.12.4b), this bound implies the bound $O(n \sum_{j=1}^{l} \mu_j^+)$ for all eigenvalues $\lambda_1, \ldots, \lambda_l$ of A. Now we recall that

$$\sum_{j=1}^{l} \mu_j^+ = n, \tag{12.5}$$

and Theorem 3.12.2 follows. □

Proof Theorem 3.12.1 We first deduce from Theorem 3.12.4 that the computation of a basis of the eigenspace of an eigenvalue λ_j of A or, equivalently, of the null space of the matrix A_j can be performed as follows:

Algorithm 3.12.1 *eigenspace computation.*

Input: Hessenberg matrix A of Theorem 3.12.4 and its eigenvalue λ_j.

Output: a basis for the eigenspace of λ_j.

Computations:

1) compute a nonsingular northwestern submatrix B of the matrix $W = P_j A_j \overline{P}_j$ where P_j and $\overline{P}_j$ are permutation matrices, and B has a sufficiently large size $(n-\mu) \times (n-\mu)$,

2) compute the matrix $-B^{-1}C$ of (12.3) and the Schur complement S of B in W,

3) compute a matrix M whose columns form a basis of the null space of S,

4) compute and output the matrix N of (12.3).

Theorem 3.12.4 fully specifies stage 1) and shows its computational cost bound.

Let us next specify stages 2)–4) and estimate their computational cost.

Stage 2). The computation of the matrix $-B^{-1}C$ amounts to the solution of μ linear systems of equations with the common coefficient matrix B. By Corollary 3.3.1, this computation uses $O((n-\mu)^2\mu) = O(n^2\mu)$ ops provided that A_j and B are Hessenberg matrices. $O(n^2\mu)$ dominates the overall cost bound at stage 2). Indeed, the multiplication of the $\mu \times (n-\mu)$ matrix E by the $(n-\mu) \times \mu$ matrix $-B^{-1}C$ requires $O((n-\mu)\mu^2)$ ops.

Stage 3). A basis for the null space of the $\mu \times \mu$ matrix S can be computed in $O(M(\mu)) = O(\mu^3)$ ops (cf. [BP94], pp. 109-110).

Stage 4). The computations amount to multiplication of $-B^{-1}C$ by M, which requires $O((n-\mu)\mu^2)$ ops.

Summarizing, we obtain a basis for the eigenspace of λ_j by using $O(n^2\mu)$ ops. Recall that $\mu \le \mu_j^+$ (by Theorem 3.12.4), sum the bounds $O(n^2\mu_j)$ over all j, recall (12.5) and obtain Theorem 3.12.1. □

Proof of Theorems 3.2.1 and 3.2.2 Theorem 3.2.1 immediately follows from Theorems 3.4.1 (cf. also Remark 3.4.1). 3.11.1 and 3.12.1. Likewise, Theorem 3.2.2 immediately follows from Theorems 3.5.5, 3.11.1 and 3.12.2. □

The reader is referred to [GL96] on the estimates for the errors of computing the eigenspace where the input eigenvalue of Algorithm 3.12.1 is given approximately within a fixed error bound. Note that λ_j are approximated according to Theorem 3.11.1, whereas the input matrix H of Algorithm 3.12.1 can be obtained exactly, by application of the algorithms of sections 3.4 and 3.5.

Remark 3.12.1 *The tests for singularity of the blocks $\lambda_j I_{\rho_s} - H_s$ and the computation of the null-space of the matrix S in Algorithm 3.12.1 can be performed numerically, with rounding the operands and outputs of each operation to a fixed (single or double) precision. Such computations should rely on the SVD computation [GL96].*

3.13 Discussion

A natural open question is whether the deterministic complexity bound of Theorem 3.2.1 can be decreased towards the lower bound $\Omega(M(n))$.

Another natural subject of further study is the Boolean complexity of the matrix eigenproblem. Based on the flowchart given in our introduction, one may obtain some reasonably good upper estimates. Indeed, the Boolean cost of stage b) (of polynomial rootfinding) has been estimated in [P95b], [P96] (cf. also [Kir,a]), yielding the optimal (up to a polylogarithmic factor) bound $O(n^2 b)$. The other stages of the flowchart, a) and c), are rational. One may perform the computations at these stages modulo several selected primes and then recover the output values by means of the Chinese remainder algorithm, thus bounding the precision of the computations in terms of the output precision p. (p is defined as the minimum precision sufficient to ensure the desired bounds on the output approximation errors.) Then we will only need to apply the well known estimates for the Boolean complexity of an arithmetic operation performed with a fixed finite precision. The remaining nontrivial problems are, of course, the estimation of the above output precision p and the Boolean complexity of the alternative approaches. We leave these open problems as a challenge for the reader.

Finally, let us briefly comment on the arithmetic complexity of the eigenproblem for the resultant matrices associated with multivariate polynomial systems of equations. Let D be a finite upper bound on the number of common roots of these equations. The $D \times D$ resultant matrices associated with such systems have certain structure and form an algebraic variety of dimension at most D in $\mathbf{C}^{D \times D}$. Equation (2.2) is extended easily to this class of matrices, whose eigenvalues equal the fixed coordinate of the common roots of the system represented in a fixed polynomial basis. This follows because, clearly, there are polynomial systems where all roots are simple and have distinct values of this coordinate (cf. Remark 3.5.3). By Corollary 3.5.1, equation (2.2) holds generically for such matrices. The extension of Corollary 3.2.1, however, depends on the solution of the important open problem of proving the bound $K_A(D) = O(D^h)$ for $h < 3$, for the matrices A of this class. Solution of this problem would have immediately given us fast solution of the polynomial system because its common roots are exactly the eigenvectors of A [AuSt88], [Ste96], [BMP98].

Appendix A

Fast Deterministic Computation of the Characteristic Polynomials of Special Matrices

For some important special classes of matrices A, there are fast deterministic algorithms for computing the characteristic polynomial of A. In particular, this is the case for Toeplitz and Toeplitz-like matrices and sparse matrices.

Theorem A.1 *[P92], [BP94]. The coefficients of the characteristic polynomial of an $n \times n$ Toeplitz, Toeplitz-like, Hankel, Hankel-like or Toeplitz-like+Hankel-like matrix can be computed in $O(n^2 \log n)$ ops.*

For some $n \times n$ matrices A, it is easy to compute $\det(b_j I_n - A)$ for fixed distinct scalars b_j, $j = 0, 1, \ldots, n$. In particular $\det A$ is immediately obtained in $n-1$ ops if one has computed either *triangular factorization (TF)*, $A = PLU$ (where P is a permutation matrix, L and U^T are lower triangular matrices and L has its main diagonal filled with ones), or the balanced recursive triangular factorization (BRTF) of A (see Definition 3.9.1). Similarly, $O(n)$ ops suffice to compute $\det(b_j I_n - A)$ if one has TF or the BRTF of $b_j I_n - A$. (Note that, by Theorem 3.9.3, there exists the BRTF of $b_j I_n - A$ if $|b_j|$ is sufficiently large.)

Theorem A.2 *Let $b_0, b_1, \ldots, b_n$ be $n+1$ distinct scalars, let A be a fixed $n \times n$ matrix, and let $F(A)$ ops suffice to compute TFs or BRTF of the matrices $b_j I_n - A$ for each j, $j = 0, \ldots, n$. Then $O(n^2) + F(A)(n+1)$ ops suffice to obtain $c_A(x) = \det(xI_n - A)$, the characteristic polynomial of A.*

Proof. $n^2 - 1$ ops suffice to compute $\det(b_j I_n - A)$ for all j if we are given TFs or BRTFs of $b_j I_n - A$, for all j. Then the characteristic polynomial $c_A(x) = \det(xI_n - A)$ is obtained by interpolation in $O(n \log^2 n)$ ops ([BP94], pp. 25-26). (For special choices of b_j, e.g. scaled roots of 1 or Chebyshev points, $O(n \log n)$ ops suffice at the interpolation stage.) □

Remark A.1 *Theorem A.2 is the main result of [R95]. We give a much shorter proof.*

Remark A.2 *$F(A)$ depends on sparsity and structure of A. For an $n \times n$ matrix A defined with its $s(n)$-separator family, we have $F(A) = O(nM(s(n)))$ [P93], [PR93]. For some important classes of sparse matrices A, we have $s(n) = \sqrt{n}$ [LT79], and then $F(A) = O(n^{3/2})$, even where we rely on the rough bound $M(n) \leq 2n^3 - n^2$, supported by* *the straightforward algorithm.*

Remark A.3 *The requirement of balancing the RTF is not needed in Theorem A.2 and can be removed.*

If we only seek the eigenvalues of a special matrix A of one of the above classes, then Theorems 3.11.1, A.1 and A.2 enable fast deterministic solution.

By applying Theorem 3.12.2 to the Frobenius matrix F satisfying (5.3), one may solve the eigenspace problem for F and A. If for a random vector $\mathbf{v}$, the Krylov matrix $K = K(A, \mathbf{v}, n)$ is nonsingular, Theorem 3.5.2 immediately gives us an extension to the solution of the entire eigenproblem for the matrix A. It is not clear, however, if one may verify nonsingularity of K at arithmetic cost which is lower than the cost estimated in Theorem 3.5.5.

Appendix B.

Approximation of the Eigenvectors by the Inverse Power Iteration

The customary numerical computation of an eigenvector $\mathbf{v}_j$ associated to a given eigenvalue λ_j goes by the Inverse Power Iteration,

$$\tilde{\mathbf{w}}^{(h)} = (\lambda_j^* I_n - A)^{-1}\mathbf{w}^{(h-1)}, \ \mathbf{w}^{(h)} = \tilde{\mathbf{w}}^{(h)}/||\tilde{\mathbf{w}}^{(h)}||, \ h = 1, 2, \ldots. \tag{B.1}$$

Here λ_j^* is a fixed approximation to λ_j, and $\mathbf{w}^{(0)}$ is a fixed normalized initial vector, $||\mathbf{w}^{(0)}|| = 1$. Iteration (B.1) stops where the vector norm $||\mathbf{w}^{(h)} - \mathbf{w}^{(h-1)}||$ becomes small enough. The vectors $\mathbf{w}^{(h)}$ converge to an eigenvector $\mathbf{v}_j$ associated with λ_j if

$$\mathbf{w}^{(0)} = \sum_{g=1}^{l} \gamma_g \mathbf{v}_g, \ \gamma_j \neq 0, j \leq l, \tag{B.2}$$

where $\mathbf{v}_g$ are some eigenvectors of A associated with eigenvalues λ_g, $g = 1, \ldots, l$, and if

$$\rho_j = |\lambda_j^* - \lambda_j|/\min_{k \neq j}|\lambda_j^* - \lambda_k| < 1; \tag{B.3}$$

furthermore, the convergence is rapid if $|\gamma_j|$ is not small and if ρ_j is small. Namely, we have the following result (cf. [GL96]):

Theorem B.1 $||\mathbf{w}^{(h)} - \mathbf{v}_j|| = O(\rho_j^h/\gamma_j)$ *as* $h \to \infty$, *under (B.1)-(B.3).*

Theorem B.2 *The computational cost of step (B.1) is* $O(n^2)$ *for a general matrix* A *reduced to Hessenberg form,* $O(n \log^2 n)$ *for a Toeplitz or Toeplitz-like matrix* A *and* $O(n)$ *for a tridiagonal matrix* A.

By Theorem B.1, the Inverse Power Iteration (B.1) converges quite fast if (B.2) and (B.3) hold, $|\gamma_j|$ is not small, and ρ_j is small. For ρ_j close or equal to 1, however, the iteration may converge very slowly or diverge. If A has n linearly independent eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$, then (B.2) holds for $l = n$ unless the vector $\mathbf{w}^{(0)}$ lies in the $(n-1)$-dimensional linear space generated by the vectors $\mathbf{v}_1, \ldots, \mathbf{v}_{j-1}, \mathbf{v}_{j+1}, \ldots, \mathbf{v}_n$. On the other hand, no general recipes are known for the choice of $\mathbf{w}^{(0)}$ that would ensure that ρ_j is small (or would just ensure this requirement even probabilistically). Moreover, ρ_j is necessarily close to 1 in the important case where several eigenvalues are clustered about λ_j . Thus, the Inverse Power Iteration should be classified as a heuristic method: it works for some (and in a certain sense for "most") of the input pairs $(A, \mathbf{w}^{(0)})$ but certainly not for all pairs.

Bibliography

[A63] S. J. Aarseth, Dynamical Evaluation of Clusters of Galaxies-I, *Mon. Not. R. Astron Soc.*, 126, 223–255, 1963.

[A85] A. W. Appel, An Efficient Program for Many-Body Simulation, *SIAM J. on Sci. Stat. Computing*, 6, 85–103, 1985.

[A86] C.R. Anderson, A Method of Local Corrections for Computing the Velocity Field due to a Distribution of Vortex Blobs, *J. Comput. Phys.*, 62, 111–123, 1986.

[AGR88] J. Ambrosiano, L. Greengard, V. Rokhlin, The Fast Multipole Method for Gridless Particle Simulation, *Computer Physics Communications*, 48, 115–125, 1988.

[AuSt88] W. Auzinger, H. Stetter, An Elimination Algorithm for the Computation of All Zeros of a System of Multivariate Polynomial Equations. In *Proc. Intern. Conf. on Numerical Math., Int. Series of Numerical Math*, **86**, 11–30, Birkhäuser, 1988.

[B84] D. Bini, Parallel Solution of Certain Toeplitz Linear Systems, *SIAM J. Comput.*, **13**, **2**, 268-276, 1984.

[B88] J. Barnes, Encounters of Disk/Halo Galaxies, *The Astrophisical J.*, 331, p. 699, 1988.

[BA80] R. R. Bitmead, B. D. O. Anderson, Asymptotically Fast Solution of Toeplitz and Related Systems of Linear Equations, *Linear Algebra Appl.*, **34**, 103-116, 1980.

[BC83] D. Bini, M. Capovani, Fast Parallel and Sequential Computations and Spectral Properties Concerning Band Toeplitz Matrices, *Calcolo,* **22** , 176-189, 1983.

[BH86] J. Barnes, P. Hut, A Hierarchical $O(N \log N)$ Force-Calculation Algorithm, *Nature*, 324, 446–449, 1986.

[BM75] A. Borodin, I. Munro, The Computational Complexity of Algebraic and Numeric Problems, *American Elsevier*, New York, 1975.

[BMP98] D. Bondyfalat, B. Mourrain, V. Y. Pan, Controlled Iterative Methods for Solving Polynomial Systems, *Proc. of ACM International Symp. on Symbolic and Algebraic Computation*, (ISSAC'98), 252-259, ACM Press, New York, 1998.

[BP86] D. Bini, V. Y. Pan, Polynomial Division and Its Computational Complexity, *J. Complexity*, **2**, 179–203, 1986.

[BP91] D. Bini, V. Y. Pan, Parallel Complexity of Tridiagonal Symmetric Eigenvalue Problem, *Proc. 2nd Annual ACM-SIAM Symp. on Discrete Algorithms*, 384-393, ACM Press, New York, and SIAM Publications, Philadelphia, 1991.

[BP94] D. Bini, V. Y. Pan, Polynomial and Matrix Computations, Volume 1: Fundamental Algorithms, *Birkhäuser, Boston*, 1994.

[BP98] D. Bini, V. Y. Pan, Computing Matrix Eigenvalues and Polynomial Zeros Where the Output Is Real, *SIAM J. on Computing*, **27**, **4**, 1099-1115, 1998.

[C73] A. J. Chorin, Numerical Study of Slightly Viscous Flaw, *Fluid Mechanics*, 57, 785–796, 1973.

[CGR88] J. Carier, L. Greengard, V. Rokhlin, A Fast Adaptive Multipole Algorithm for Particle Simulation, *SIAM J. Sci. Stat. Comput.*, 9, 4, 669–686, 1988.

[D83] J. M. Dawson, Particle Simulation of Plasmas, *Rev. Mod. Physics*, 55, 403–447, 1983.

[DL78] R. A. Demillo, R. J. Lipton, A Probabilistic Remark on Algebraic Program Testing, *Information Process. Letters*, **7**, **4**, 193–195, 1978.

[FF63] D. K. Faddeev, V. N. Faddeeva, Computational Methods of Linear Algebra, *W.H. Freeman*, San Francisco, 1963.

[FHR93] T. Fink, G. Heinig, K. Rost, An Inversion Formula and Fast Algorithms for Cauchy-Vandermonde Matrices, *Linear Algebra Appl.*, 183, 179–191, 1993.

[G88] L. Greengard, The Rapid Evaluation of Potential Fields in Particle Systems, *MIT Press*, Cambridge, Mass, 1988.

[Ger87] A. Gerasoulis, A Fast Algorithm for the Multiplication of Generalized Hilbert Matrices with Vectors, *Math. Comp.*, 50, 181, 179–188, 1987.

[GIKT95] A. George, K. Ikramov, A. N. Krivoshapova, W.-P. Tang, A Finite Procedure for the Tridiagonalization of a General Matrix, *SIAM J. on Matrix Analysis & Applics.*, **16**, **2**, 377-387, 1995.

[GL96] G. H. Golub, C. F. Van Loan, Matrix Computations, *Johns Hopkins Univ. Press*, Baltimore, Maryland, 1996 (third edition).

[Go94] S. Goedecker, Remark on Algorithms to Find Roots of Polynomials, *SIAM J. on Sci. Computing*, **15**, 1094-1063, 1994.

[GO94] I. Gohberg, V. Olshevsky, Complexity of Multiplication with Vectors for Structured Matrices, *Linear Algebra and Its Applications,* **202**, 163–192, 1994.

[GR87] L. Greengard, V. Rokhlin, A Fast Algorithm for Particle Simulation, *J. Comput. Physics*, 73, 2, 325–348, 1987.

[HE81] R. W. Hockney, J. W. Eastwood, Computer Simulation Using Particles, *McGraw-Hill*, New York, 1981.

[Hei95] G. Heinig, Inversion of Generalized Cauchy Matrices and the Other Classes of Structured Matrices, *Linear Algebra for Signal Processing, IMA Volume in Math. and Its Applications*, **69**, 95-114, Springer, 1995.

[Hig96] N. J. Higham, Accuracy and Stability in Numerical Analysis, *SIAM Publications*, Philadelphia, 1996.

[K,a] I. Kaporin, A Practical Algorithm for Faster Matrix Multiplication, *Numerical Linear Algebra with Applications*, accepted.

[Kal95] E. Kaltofen, Analysis of Coppersmith's Block Wiedemann Algorithm for the Parallel Solution of Sparse Linear Systems, *Math. of Comput.*, **64**, **210**, 777-806, 1995.

[K-G85] W. Keller-Gehrig, Fast Algorithms for Characteristic Polynomial, *Theoretical Computer Science*, **36**, 309-317, 1985.

[Kir,a] P. Kirrinnis, Partial Fraction Decomposition in $C(z)$ and Simultaneous Newton's Iteration for Factorization in $C[Z]$, *J. of Complexity*, to appear.

[KKM79] T. Kailath, S.Y. Kung, M. Morf, Displacement Ranks of Matrices and Linear Equations, *J. Math. Anal. Appl.*, **68**, **2**, 395-407, 1979.

[KO96] T. Kailath, V. Olshevsky, Displacement Structure Approach to Discrete-Trigonometric-Transform Based Preconditioners of G. Strang Type and of T. Chan Type, *Calcolo*, **33**, 1996.

[KP91] E. Kaltofen, V. Y. Pan, Processor Efficient Parallel Solution of Linear Systems over an Abstract Field, *Proc. 3rd Ann. ACM Symp. on Paralled Algorithms and Architectures*, 180-191, ACM Press, New York, 1991.

[KS91] E. Kaltofen, B. D. Saunders, On Wiedemann's Method for Solving Sparse Linear Systems, *Proc. AAECC-5, Lecture Notes in Computer Science*, **536**, 29–38, Springer, Berlin, 1991.

[KW92] J. Kuczyński, H. Woźniakovski, Estimating the Largest Eigenvalue by the Power and Lanczos Algorithms with Random Start, *SIAM J. on Matrix Analysis and Applications*, **13**, 1094-1122, 1992.

[LT79] R. J. Lipton, R. E. Tarjan, A Separator Theorem for Planar Graphs, *SIAM J. on Applied Math.*, **36**, **2**, 177-189, 1979.

[M80] M. Morf, Doubling Algorithms for Toeplitz and Related Equations, *Proc. IEEE Internet. Conf. on ASSP*, 954-959, IEEE Computer Society Press, 1980.

[MP96] B. Mourrain, V. Y. Pan, Multidimensional Structured Matrices and Polynomial Systems, *Calcolo*, **33**, 389-401, 1996.

[MP97] B. Mourrain, V. Y. Pan, Solving Special Polynomial Systems by Using Structured Matrices and Algebraic Residues, *Proc. of the AMS-SIAM Conference on Foundation of Computational Math.* (F. Cucker and M. Shub editors), Rio-de-Janeiro, pp. 287-304, Springer, January 1997,

[MP98] B. Mourrain, V. Y. Pan, Asymptotic Acceleration of Solving Multivariate Polynomial Systems of Equations, *Proc. 30th Ann. ACM Symp. on Theory of Computing,* 488-496, ACM Press, New York, 1998.

[ODR89] S.T. O'Donnell, V. Rokhlin, A Fast Algorithm for Numerical Evaluation of Conformal Mappings, *SIAM J. Sci. Stat. Comput.*, 10, 3, 475–487, 1989.

[OP98] V. Olshevsky, V. Y. Pan, A Unified Superfast Algorithm for Boundary Rational Tangential Interpolation Problem and for Inversion and Factorization of Dense Structured Matrices, *Proc. 39th Annual IEEE Symposium on Foundations of Computer Science,* IEEE Computer Society Press, 1998.

[OS88] A.M. Odlyzko, A. Schönhage, Fast Algorithms for Multiple Evaluations of the Riemann Zeta Function, *Trans. Am. Math. Soc.*, 309, 2, 797–809, 1988.

[P92] V. Y. Pan, Parametrization of Newton's Iteration for Computations with Structured Matrices and Applications, *Computers & Math. (with Applics.)*, **24, 3**, 61–75, 1992.

[P93] V. Y. Pan, Parallel Solution of Sparse Linear and Path Systems, in *Synthesis of Parallel Algorithms* (J. H. Reif editor), chapter 14, pp. 621-678. Morgan Kaufmann publishers, San Mateo, California, 1993.

[P95] V. Y. Pan, An Algebraic Approach to Approximate Evaluation of a Polynomial on a Set of Real Points, *Advances in Computational Mathematics*, 3, 41–58, 1995.

[P95b] V. Y. Pan, Optimal (up to Polylog Factors) Sequential and Parallel Algorithms for Approximating Complex Polynomial Zeros, *Proc. 27th Ann. ACM Symp. on Theory of Computing*, 741-750, ACM Press, New York, 1995.

[P96] V. Y. Pan, Optimal and Nearly Optimal Algorithms for Approximating Polynomial Zeros, *Computers & Math. (with Applications)*, **31**, **12**, 97–138, 1996.

[P97] V. Y. Pan, Solving a Polynomial Equation: Some History and Recent Progress, *SIAM Review*, **39**, **2**, 187-220, 1997.

[P98] V. Y. Pan, Approximate Polynomial Gcds, Padé Approximation, Polynomial Zeros, and Bipartite Graphs, *Proc. 9th Ann. ACM-SIAM Symp. on Discrete Algorithms,* 68-77, *ACM Press*, New York, and *SIAM Publications*, Philadelphia, 1998.

[Par80] B. N. Parlett, The Symmetric Eigenvalue Problem, Prentice-Hall, 1980.

[Par92] B. N. Parlett, Reduction to Tridiagonal Form and Minimal Realizations, *SIAM J. Matrix Anal. Appl.*, **13**, **2**, 567–593, 1992.

[PACLS98] V. Y. Pan, M. AbuTabanjeh, Z. Chen, E. Landowne, A. Sadikou, New Transformations of Cauchy Matrices and Trummer's Problem, *Computer and Math. (with Applics.)*, **35**, **12**, 1-5, 1998.

[PACPS98] V. Y. Pan, M. AbuTabanjeh, Z. Chen, S. Providence, A. Sadikou, Transformations of Cauchy Matrices for Trummer's Problem and a Cauchy-like Linear Solver, *Proc. of 5th Annual International Symposium on Solving Irregularly Structured Problems in Parallel, (Irregular98)*, (A. Ferreira, J. Rolim, H. Simon, S.-H. Teng Editors), *Lecture Notes in Computer Science*, **1457**, 274-284, Springer, 1998.

[PC99] V. Y. Pan, Z. Chen, Approximate Real Polynomial Division via Approximate Inversion of Real Triangular Toeplitz Matrices, *Applied Math. Letters*, **12**, 1-2, 1999.

[PC99a] V. Y. Pan, Z. Chen, The Complexity of the Algebraic Eigenproblem, *Proc. 31st Annual ACM Symposium on Theory of Computing*, 507-516, ACM Press, New York, 1999.

[PLST93] V.Y. Pan, E. Landowne, A. Sadikou, O. Tiga, A New Approach to Fast Polynomial Interpolation and Multipoint Evaluation, *Computers & Math. (with Applications)*, 25, 9, 25–30, 1993.

[PR93] V. Y. Pan, J. Reif, Fast and Efficient Parallel Solution of Sparse Linear Systems, *SIAM J. on Computing* , **22, 6**, 1227-1250, 1993.

[PSD70] P. Penfield Jr., R. Spencer, S. Duinker, Tellegen's Theorem and Electrical Networks, *MIT Press*, Cambridge, Massachusetts, 1970.

[PZHY97] V. Y. Pan, A. Zheng, X. Huang, Y. Yu, Fast Multipoint Polynomial Evaluation and Interpolation via Computation with Structured Matrices, *Annals of Numerical Math.*, 4, 483–510, 1997.

[Q94] M. J. Quinn, Parallel Computing: Theory and Practice, *McGraw-Hill*, New York, 1994.

[R85] V. Rokhlin, Rapid Solution of Integral Equations of Classical Potential Theory, *J. Comput. Physics*, 60, 187–207, 1985.

[R87] J. Renegar, On the Worst-case Arithmetic Complexity of Approximating the Zeros of Polynomials, *J. of Complexity,* **3**, 90-113, 1987.

[R88] V. Rokhlin, A Fast Algorithm for the Discrete Laplace Transformation, *J. of Complexity*, 4, 12–32, 1988.

[R95] J. H. Reif, Efficient Parallel Solution of Sparse Eigenvalue and Eigenvector Problems, *Proc. of Ann. IEEE Symp. on Foundations of Computer Science,* 123-132, IEEE Computer Soc. Press, 1995.

[S80] J. T. Schwartz, Fast Probabilistic Algorithms for Verification of Polynomial Identities, *J. of ACM,* **27**, **4**, 701–717, 1980.

[Ste96] H. J. Stetter, Eigenproblems Are at the Heart of Polynomial System Solving, *ACM Bulletin,* **30, 4**, ACM Press, New York, 22-25, 1996.

[Str69] V. Strassen, Gaussian Elimination Is Not Optimal, *Numer. Math.,* **13**, 354–356, 1969.

[T96] E. E. Tyrtyshnikov, A Unified Approach to Old and New Theorems on Distribution and Clustering, *Linear Algebra and Its Applications*, **232**, 1-43, 1996.

[VDR89] L. van Dommelen, E. A. Rundensteiner, Fast Adaptive Summation of Point Forces in the Two-Dimentional Poisson Equation, *J. Comp. Physics*, 83, 126–147, 1989.

[Wi86] D. H. Wiedemann, Solving Sparse Linear Equations Over Finite Fields, *IEEE Trans. on Inf. Theory*, **IT-32, 1**, 54-62, 1986.

[Wil65] J.H. Wilkinson, The Algebraic Eigenvalue Problem, *Clarendon Press*, Oxford, 1965.

[Y76] D. Y. Yun, On Square-free Decompositon Algorithm, *Proc. ACM Symp. on Symbolic and Algebraic Computation*, 26-35, ACM Press, New York, 1976.

[Z79] R. E. Zippel, Probabilistic Algorithms for Sparse Polynomials, *Proc. EUROSAM 79, Lecture Notes in Computer Science*, **72**, 216–226, Springer, Berlin, 1979.

ndt