

# CONTEMPORARY MATHEMATICS

323

## Fast Algorithms for Structured Matrices: Theory and Applications

AMS-IMS-SIAM Joint Summer Research Conference  
on Fast Algorithms in Mathematics,  
Computer Science and Engineering  
August 5–9, 2001

Mount Holyoke College, South Hadley,  
Massachusetts

Vadim Olshevsky  
Editor



American Mathematical Society  
Society for Industrial and Applied Mathematics



# Fast Algorithms for Structured Matrices: Theory and Applications

*This page intentionally left blank*

# CONTEMPORARY MATHEMATICS

---

323

## Fast Algorithms for Structured Matrices: Theory and Applications

AMS-IMS-SIAM Joint Summer Research Conference  
on Fast Algorithms in Mathematics,

Computer Science and Engineering

August 5–9, 2001

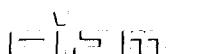
Mount Holyoke College, South Hadley,  
Massachusetts

Vadim Olshevsky  
Editor



American Mathematical Society  
Providence, Rhode Island

Society for Industrial and Applied Mathematics  
Philadelphia, PA



## Editorial Board

Dennis DeTurck, managing editor

Andreas Blass    Andy R. Magid    Michael Vogelius

The AMS-IMS-SIAM Joint Summer Research Conference on “Fast Algorithms in Mathematics, Computer Science and Engineering” was held at Mount Holyoke College, South Hadley, Massachusetts, August 5–9, 2001, with support from the National Science Foundation, grant DMS 9973450.

2000 *Mathematics Subject Classification*. Primary 68Q25, 65Y20, 65F05, 65F10, 65G50.  
65M12, 15A57, 15A18, 47N70, 47N40.

SIAM is a registered trademark

---

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

---

### Library of Congress Cataloging-in-Publication Data

AMS-IMS-SIAM Joint Summer Research Conference on Fast Algorithms in Mathematics, Computer Science, and Engineering (2001 : Mount Holyoke College)

Fast algorithms for structured matrices : theory and applications : AMS-IMS-SIAM Joint Summer Research Conference on Fast Algorithms in Mathematics, Computer Science, and Engineering, August 5–9, 2001, Mount Holyoke College, South Hadley, Massachusetts / Vadim Olshevsky, editor.

p. cm. — (Contemporary mathematics, ISSN 0271-4132 ; 323)

Includes bibliographical references.

ISBN 0-8218-3177-1 (acid-free paper) — ISBN 0-89871-543-1 (acid-free paper)

1. Matrices—Congresses. 2. Fourier transformations—Congresses. 3. Algorithms—Congresses. I. Olshevsky, Vadim, 1961-. II. Title. III. Contemporary mathematics (American Mathematical Society); v. 323.

QA188.J65 2001

512.9'434—dc21

2003041790

---

**Copying and reprinting.** Material in this book may be reproduced by any means for educational and scientific purposes without fee or permission with the exception of reproduction by services that collect fees for delivery of documents and provided that the customary acknowledgment of the source is given. This consent does not extend to other kinds of copying for general distribution, for advertising or promotional purposes, or for resale. Requests for permission for commercial use of material should be addressed to the Acquisitions Department, American Mathematical Society, 201 Charles Street, Providence, Rhode Island 02904-2294, USA. Requests can also be made by e-mail to [reprint-permission@ams.org](mailto:reprint-permission@ams.org).

Excluded from these provisions is material in articles for which the author holds copyright. In such cases, requests for permission to use or reprint should be addressed directly to the author(s). (Copyright ownership is indicated in the notice in the lower right-hand corner of the first page of each article.)

© 2003 by the American Mathematical Society. All rights reserved.

The American Mathematical Society retains all rights  
except those granted to the United States Government.

Printed in the United States of America.

⊕ The paper used in this book is acid-free and falls within the guidelines  
established to ensure permanence and durability.

Visit the AMS home page at <http://www.ams.org/>

## Contents

Foreword	vii
Pivoting for Structured Matrices and Rational Tangential Interpolation VADIM OLSHEVSKY	1
Inversion of Toeplitz-Plus-Hankel Matrices with Arbitrary Rank Profile GEORG HEINIG	75
A Lanczos-type Algorithm for the QR Factorization of Cauchy-like Matrices DARIO FASINO AND LUCA GEMIGNANI	91
Fast and Stable Algorithms for Reducing Diagonal Plus Semiseparable Matrices to Tridiagonal and Bidiagonal Form DARIO FASINO, NICOLA MASTRONARDI, AND MARC VAN BAREL	105
A Comrade-Matrix-Based Derivation of the Eight Versions of Fast Cosine and Sine Transforms ALEXANDER OLSHEVSKY, VADIM OLSHEVSKY, AND JUN WANG	119
Solving Certain Matrix Equations by Means of Toeplitz Computations: Algorithms and Applications DARIO A. BINI, LUCA GEMIGNANI, AND BEATRICE MEINI	151
A Fast Singular Value Algorithm for Hankel Matrices FRANKLIN T. LUK AND SANZHENG QIAO	169
A Modified Companion Matrix Method Based on Newton Polynomials D. CALVETTI, L. REICHEL, AND F. SGALLARI	179
A Fast Direct Method for Solving the Two-dimensional Helmholtz Equation, with Robbins Boundary Conditions J. HENDRICKX, RAF VANDEBRIL, AND MARC VAN BAREL	187
Structured Matrices in Unconstrained Minimization Methods CARMINE DI FIORE	205
Computation of Minimal State Space Realizations in Jacobson Normal Form NAOHARU ITO, WILAND SCHMALE, AND HARALD K. WIMMER	221
High Order Accurate Particular Solutions of the Biharmonic Equation on General Regions ANITA MAYO	233

A Fast Projected Conjugate Gradient Algorithm for Training Support Vector Machines TONG WEN, ALAN EDELMAN, AND DAVID GORSICH	245
A Displacement Approach to Decoding Algebraic Codes V. OLSHEVSKY AND M. AMIN SHOKROLLAHI	265
Some Convergence Estimates for Algebraic Multilevel Preconditioners MATTHIAS BOLLHÖFER AND VOLKER MEHRMANN	293
Spectral Equivalence and Matrix Algebra Preconditioners for Multilevel Toeplitz Systems: A Negative Result D. NOUTSOS, S. SERRA CAPIZZANO, AND P. VASSALOS	313
Spectral Distribution of Hermitian Toeplitz Matrices Formally Generated by Rational Functions WILLIAM F. TRENCH	323
From Toeplitz Matrix Sequences to Zero Distribution of Orthogonal Polynomials DARIO FASINO AND STEFANO SERRA CAPIZZANO	329
On Lie Algebras, Submanifolds and Structured Matrices KENNETH R. DRIESSEL	341
Riccati Equations and Bitangential Interpolation Problems with Singular Pick Matrices HARRY DYM	361
Functions with Pick Matrices having Bounded Number of Negative Eigenvalues V. BOLOTNIKOV, A. KHEIFETS, AND L. RODMAN	393
One-dimensional Perturbations of Selfadjoint Operators with Finite or Discrete Spectrum YU. M. ARLINSKIĬ, S. HASSI, H. S. V. DE SNOO, AND E. R. TSEKANOVSKIĬ	419

## Foreword

Perhaps the most widely known example of fast algorithms is the *fast Fourier transform* (FFT) algorithm. Its importance is widely acknowledged and nicely described in numerous papers and monographs, e.g., as follows: “*The fast Fourier transform (FFT) is one of the truly great computational developments of this century. It has changed the face of science and engineering so that it is not an exaggeration to say that life as we know it would be very different without FFT*” (Charles Van Loan, *Computational Frameworks for the Fast Fourier Transform*, SIAM Publications, 1992). There are many different mathematical languages which can be used to derive and describe the FFT, and the “*structured matrices language*” is one of them, yielding the following interpretation. Though the usual matrix-vector multiplication uses  $n(2n - 1)$  arithmetic operations, the *special structure* of the discrete Fourier transform matrix allows us to reduce the latter complexity to the nearly linear cost of  $O(n \log n)$  operations. The practical importance of such a dramatic speed-up is impossible to overestimate; even for moderately sized problems one can compute the result hundreds of times faster.

This is a model example showing why reducing the computational burden via structure exploitation is an important issue in many applied areas. Thus, it is not surprising that in recent years the design of fast algorithms for structured matrices has become an increasingly important activity in a diverse variety of branches of the exact sciences. Unfortunately, until recently there was not much interaction between the different branches. There were no comprehensive meetings bringing together “all interested parties,” and no cross-disciplinary publishing projects. Such situations caused several disadvantages.

First, there clearly was a certain parallelism, and several algorithms have independently been rediscovered in different areas. For example, the Chebyshev continuous fraction algorithm for interpolation, the Stiltjes procedure for generating orthogonal polynomials, the Lanczos algorithm for computing the eigenvalues of a symmetric matrix, and the Berlekamp-Massey algorithm for decoding of BCH codes are closely related. Another example: the classical Nevanlinna algorithm for rational passive interpolation and the Darlington procedure for passive network synthesis are variations on the same theme.

The second disadvantage of the lack of cross-disciplinary interaction is that it was not clear that the research efforts in different branches are part of what one can call a “full research cycle,” starting from an actual application, through developing deep theories, to the design of efficient algorithms and their implementation. Researchers in different branches often had somewhat narrower focuses. Electrical engineers used structured matrices to efficiently solve applied problems.

Mathematicians exploited the structure to obtain elegant solutions for various fundamental problems. Computer scientists utilized the structure to speed up related algorithms and to study the corresponding complexity problems. Numerical analysts took advantage of the structure to improve accuracy in many cases.

In recent years such an unfortunate situation has changed. We had a number of cross-disciplinary conferences in Santa Barbara (USA, Aug. 1996), Cortona (Italy, Sept. 1996 and Sept. 2000), Boulder (USA, July 1999), Chemnitz (Germany, Jan. 2000), South Hadley (USA, Aug. 2001). In fact, it was the "cross-fertilization" atmosphere of the South Hadley meeting that suggested the idea to pursue this publishing project. We hope it demonstrates the following two points. First, the approaches, ideas and techniques of engineers, mathematicians, and numerical analysts nicely complement each other, and despite their differences in techniques and agendas they can be considered as important parts of a joint research effort. Secondly, the theory of structured matrices and design of fast algorithms for them seem to be positioned to bridge several diverse fundamental and applied areas.

The volume contains twenty-two survey and research papers devoted to a variety of theoretical and practical aspects of design of fast algorithms for structured matrices and related issues. It contains a number of papers on direct fast algorithms and also on iterative methods. The convergence analysis of the latter requires studying spectral properties of structured matrices. The reader will find here several papers containing various affirmative and negative results in this direction. The theory of rational interpolation is one of the excellent sources providing intuition and methods to design fast algorithms. This volume contains several computational and theoretical papers on the topic. There are several papers on new applications of structured matrices, e.g., the design of fast decoding algorithms, computing state-space realizations, relations to Lie algebras, unconstrained optimization, solving matrix equations, etc.

We hope that the reader will enjoy a plethora of different problems, different focuses, and different methods that all contribute to one unified theory of fast algorithms for structured matrices and related theoretical issues.

Vadim Olshevsky

Department of Mathematics  
University of Connecticut  
Storrs, CT 06269, USA

# Pivoting for Structured Matrices and Rational Tangential Interpolation

Vadim Olshevsky

**ABSTRACT.** Gaussian elimination is a standard tool for computing triangular factorizations for general matrices, and thereby solving associated linear systems of equations. As is well-known, when this classical method is implemented in finite-precision-arithmetic, it often fails to compute the solution accurately because of the accumulation of small roundoffs accompanying each elementary floating point operation. This problem motivated a number of interesting and important studies in modern numerical linear algebra; for our purposes in this paper we only mention that starting with the breakthrough work of Wilkinson, several *pivoting techniques* have been proposed to stabilize the numerical behavior of Gaussian elimination.

Interestingly, matrix interpretations of many known and new algorithms for various applied problems can be seen as a way of computing triangular factorizations for the associated *structured matrices*, where different patterns of structure arise in the context of different physical problems. The special structure of such matrices [e.g., Toeplitz, Hankel, Cauchy, Vandermonde, etc.] often allows one to *speed-up* the computation of its triangular factorization, i.e., to efficiently obtain *fast implementations* of the Gaussian elimination procedure. There is a vast literature about such methods which are known under different names, e.g., fast Cholesky, fast Gaussian elimination, generalized Schur, or Schur-type algorithms. However, without further improvements they are efficient fast implementations of a numerically inaccurate [for indefinite matrices] method.

In this paper we survey recent results on the fast implementation of various pivoting techniques which allowed us to improve numerical accuracy for a variety of fast algorithms. This approach led us to formulate new more accurate numerical methods for factorization of general and of  $J$ -unitary rational matrix functions, for solving various tangential interpolation problems, new Toeplitz-like and Toeplitz-plus-Hankel-like solvers, and new divided differences schemes. We believe that similar methods can be used to design accurate fast algorithm for the other applied problems and a recent work of our colleagues supports this anticipation.

---

1991 *Mathematics Subject Classification*. Primary 65F30, 15A57; Secondary 65Y20.

*Key words and phrases.* Fast algorithms, Pivoting, Structured matrices, Interpolation.

This work was supported in part by NSF contracts CCR 0098222 and 0242518.

## 1. Motivation and a first example of structure: a Vandermonde matrix

The standard method for solving a  $n \times n$  linear system of simultaneous equations, Gaussian elimination [GE], requires  $O(n^3)$  arithmetic operations per system. However, this computational burden is certainly too expensive if the size  $n$  of the coefficient matrix is large, so often the use of GE can be impractical. Fortunately, many linear equations appearing in applications often have a certain *structure*, introduced by a particular physical phenomenon, and this structure can be exploited to speed-up the computation. For example, a Vandermonde linear system of equations,

$$(1.1) \quad V(x) \cdot a = f, \quad \text{where} \quad V(x) = [x_k^{j-1}]_{1 \leq k, j \leq n}$$

can be rapidly solved in only  $O(n^2)$  operations via the *fast* Björck-Pereyra [BP] algorithm [BP70], [GVL96]. Of course, gain in speed is a very important factor from the practical point of view, but there are several other features that we may wish the algorithm to satisfy, such as amenability to parallel implementations, memory requirements and traffic, etc. In floating point arithmetic implementations, where the roundoff errors are present, the crucial factor that makes an algorithm practical is its *numerical accuracy*.

The accuracy of general purpose, i.e., *structure-ignoring* algorithms, such as GE, has been extensively studied over the last few decades, and for such algorithms we now have various so-called *stabilizing techniques* that improve the numerical accuracy of these algorithms. As a matter of fact, such techniques are based on the fact that a certain algorithm usually can be implemented in a variety of mathematically equivalent ways. So we often can use a certain heuristic to choose an implementation providing the *computed result* with a better accuracy.

In contrast, the analysis of accuracy for *fast algorithms*, viz., those that *exploit the structure* to speed-up the computation, has received much less attention. Moreover, we know many fast algorithms that are less accurate as compared to their general purpose counterparts. Of course, it would be too hasty to conclude that there is a certain “*original sin*” associated with all fast algorithms, and the explanation can be much simpler: at the moment the interest in the numerical behavior of fast algorithms is quite recent, so fewer stabilizing techniques are available. In this paper we survey recent progress in the design of fast implementations for one well-known technique of enhancing accuracy: *pivoting*.

To motivate the further discussions let us return to the special system of linear equations in (1.1), and observe that any reordering of the rows of  $V(x)$  does not destroy the Vandermonde structure, and that it is equivalent to the reordering of the nodes  $\{x_k\}$ . Therefore, for each available algorithm, we can solve system (1.1) in a variety of mathematically equivalent, but *numerically different* ways, by preceding this algorithm with a certain permutation of  $\{x_k\}$ . Consider the following two particular orderings:

- Monotonic ordering,

$$x_1 < x_2 < \dots < x_n,$$

- Leja ordering,

$$(1.2) \quad |x_1| = \max_{1 \leq j \leq n} |x_j|, \\ \prod_{i=1}^{k-1} |x_k - x_i| = \max_{k \leq j \leq n} \prod_{i=1}^{k-1} |x_j - x_i| \quad \text{for } 2 \leq k \leq n : .$$

Condition number of $V(x)$	Monotonic ordering		Leja ordering	
	GE	BP	GE	BP
2.36e+06	3.1e-08	9.7e-09	1.9e-10	2.3e-10

TABLE 1. *Residual error for nodes equidistant in (-1,1).*

Condition number of $V(x)$	Monotonic ordering		Leja ordering	
	GE	BP	GE	BP
1.53e+12	7.4e-05	2.7e-05	9.5e-05	6.5e-05

TABLE 2. *Residual error for nodes equidistant in (0,1).*

We used MATLAB<sup>1</sup> to solve two  $15 \times 15$  Vandermonde systems of the form (1.1), corresponding to the following two node configurations:

- equidistant in  $(-1,1)$ ,
- equidistant in  $(0,1)$ ,

and for the right-hand-side  $f = [1 \ -1 \ 1 \ -1 \ \dots]^T$ . For each of the two systems we tried two above orderings of  $\{x_k\}$ . For each of these four cases we applied GE and BP algorithms to compute the solution  $\hat{a}$ ; this computed solution is, of course, inaccurate, because of the roundoffs accompanying each elementary step of computation. The results of the measurement of the residual error  $\|V(x)\hat{a} - f\|_\infty$  are shown in Tables 1 and 2.

A closer look at the data in each table allows us to make the following two observations.

- **General matrices. Leja ordering.** The data in Table 1 and in numerous other examples suggest that for the nodes of both signs,  $x_k \in (-1, 1)$ , the use of Leja ordering is a stabilizing technique for the both GE and BP algorithms, i.e., it provides a smaller size for the residual errors. This confirms our experience and the computational experience of many others that Leja ordering enhances the accuracy of various algorithms for Vandermonde matrices [Hig90], [R90b], [RO91], [GK93], [GO94a], [GO97].
- **Special matrices. Monotonic ordering.** Before analyzing the data in Table 2 observe that the coefficient matrix here differs from the one in Table 1 which implies different conditioning, different sizes of the solution, etc. Therefore we should just ignore any differences between the two above tables, and our analysis here should be aimed only at the data in Table 2. These data indicate that for the special class of Vandermonde matrices with positive nodes,  $x_k \in (0, 1)$ , there is no need for Leja ordering, and monotonic ordering provides about the same accuracy.

The explanation for these two observations is simple enough to be briefly stated next.

- **General matrices. Partial pivoting.** The result in Table 2 is explained by the observation [perhaps first noted in [Hig90]] that (1.2) is nothing

---

<sup>1</sup>MATLAB is a trademark of The MathWorks Inc.

else but *partial pivoting*, *efficiently implemented* for a Vandermonde matrix.

- **Total positivity. Avoidance of pivoting.** The justification for the results in Table 2 is the fact that there are classes of matrices, in this case totally positive matrices, for which it is advantageous not to pivot [totally positive matrices are those for which the determinant of every submatrix is positive, see classic monographs [GK50], [K72] for a list of applications].

For the BP algorithms these two conclusions will be more transparently presented later in Sec. 9. For the GE procedure these two explanations are more explicit in the next section where we set the notation, and briefly survey several tutorial facts on pivoting for structure-ignoring general algorithms. Having this background, in the rest of the paper we shall present several similar stabilizing techniques for various other kinds of structure, and extend them to several other fast algorithms.

## 2. Ignoring structure: Pivoting to improve the accuracy of Gaussian elimination (GE)

**2.1. Gaussian elimination.** The first step of the GE procedure applied to a matrix  $R_1$  can be described as a factorization of the following form. Let  $R_1 := R$ , then

$$(2.1) \quad R_1 \triangleq \begin{bmatrix} d_1 & u_1 \\ l_1 & R_{22}^{(1)} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{1}{d_1}l_1 & I \end{bmatrix} \cdot \begin{bmatrix} d_1 & u_1 \\ 0 & R_2 \end{bmatrix},$$

where the matrix

$$(2.2) \quad R_2 = R_{22}^{(1)} - \frac{1}{d_1}l_1u_1$$

is called the *Schur complement* of (1,1) entry  $d_1$  in the matrix  $R_1$ . This step provides the first column  $\begin{bmatrix} 1 \\ \frac{1}{d_1}l_1 \end{bmatrix}$  of the lower-triangular factor  $L$  and the first row  $\begin{bmatrix} d_1 & u_1 \end{bmatrix}$  of the upper triangular factor  $U$  in the LU factorization of  $R_1 = LU$ . Recursively proceeding to factorize the Schur complement  $R_2$ , after  $n-1$  steps we obtain the whole LU factorization.

Now, it is well-known, see, e.g., [SB80] or [Hig96], that in the standard model of floating point arithmetic,

$$(2.3) \quad fl\{x \text{ op } y\} = (x \text{ op } y)(1 + \delta), \quad |\delta| \leq u,$$

where  $op \in \{+, -, *, /\}$ ,  $fl\{x \text{ op } y\}$  denotes the *computed result*, and  $u$  is the *unit roundoff*,

$$u \approx \begin{cases} 10^{-8} & \text{in single precision} \\ 10^{-16} & \text{in double precision} \end{cases}$$

we have the following error bounds:

- **Factorization error.** The triangular factors  $\widehat{L}$  and  $\widehat{U}$  computed by Gaussian elimination satisfy

$$(2.4) \quad \widehat{L}\widehat{U} = R + \Delta R, \quad |\Delta R| \leq \gamma_n |\widehat{L}| |\widehat{U}|, \quad \text{where} \quad \gamma_n = \frac{nu}{1-nu}.$$

where the operations of taking the absolute value and of comparison are understood *componentwise*.

- **Backward error.** The use of the computed triangular factors  $\widehat{L}$  and  $\widehat{U}$  to solve the associated linear system  $R\hat{a} = f$  via forward and back-substitution produces a *computed solution*  $\hat{a}$  that is the exact solution of a nearby system  $(R + \Delta R)\hat{a} = f$ , where

$$(2.5) \quad |\Delta R| \leq 2\gamma_n |\widehat{L}| |\widehat{U}|.$$

- **Residual error.** Finally,

$$(2.6) \quad |f - R\hat{a}| \leq 2\gamma_n |\widehat{L}| |\widehat{U}| |\hat{a}|.$$

All three bounds involve the product  $|\widehat{L}| |\widehat{U}|$ , which can be quite large, indicating the potential numerical instability, in general, of the Gaussian elimination procedure. However, these bounds give a clue as how to numerically stabilize this classical algorithm. The standard way to deal with the often large size of

$$(2.7) \quad g_n = \frac{\|\widehat{L}\| |\widehat{U}| \|_\infty}{\|R\|_\infty}$$

is to exploit the freedom of row interchanges for  $R_k$  in (2.1) before the  $k$ -th step of elimination. Such techniques are called *pivoting* techniques, two of which are discussed next.

**2.2. Partial or complete pivoting? Practical experience vs. error bounds.** In this short subsection we briefly recall widely known heuristic arguments suggesting that while complete pivoting has better error bounds, nevertheless faster partial pivoting usually provides the same accuracy and is preferable. Though reasonings here concern with slow structure-ignoring algorithms only, but as we shall conjecture in Sec. 7.6 below the situation with fast transformation-and-pivoting algorithms exploiting the Toeplitz structure is analogous.

The *partial pivoting* technique performs the elimination on a permuted version of the coefficient matrix,  $PR = LU$ , where the permutation  $P$  is built recursively:  $P = P_{n-1} \cdots P_1$ . More specifically, an elementary permutation  $P_k$  is chosen to bring, at the  $k$ -th step, the maximal magnitude entry of the  $k$ -th column to the pivotal (k,k) position. This guarantees that the entries of the computed lower triangular factor  $\widehat{L}$  are all less than 1, and loosely speaking, that the quantity  $g_n$  is usually of the order of unity, resulting in a small backward error (2.5) for GE *in practice*. The proviso “*in practice*” means that though partial pivoting has a good record of accurate performance, there are several examples for which it is not sufficient, and the backward error can be still large [see, e.g., [\[Hig96\]](#) and the references therein]. In these cases *complete pivoting* will do better: in this technique one recursively chooses a permutation matrix  $P$  in  $PR = LU$  to bring, at the  $k$ -th step, the maximal magnitude entry in the *whole matrix* [not just in the first column] to the pivotal (k,k) position. However, complete pivoting requires significantly more comparisons, so partial pivoting is the stabilizing technique of the choice in most commercial and shareware packages, e.g., in MATLAB, LAPACK, etc. This choice is based on the fact that the size of (2.7) [so also of the backward error] is almost invariably small in practice for partial pivoting.

Here we might mention an interesting remark of W.M.Kahan [\[K80\]](#): “*Intolerable pivot-growth [with partial pivoting] is a phenomenon that happens only to numerical analysts who are looking for that phenomenon.*”

Moreover, in a recent monograph [Hig96] N.J.Higham states that “*although there are practically occurring matrices for which partial pivoting yields a moderately large, or even exponentially large, growth factor, the growth factor is almost invariably found to be small. Explaining this fact remains one of the major unsolved problems in numerical analysis.*”

Interestingly, the importance of a proper balance between practical experience and error bounds [such *a-priori* bounds are “*often impractically large*” [W71]] was recognized already in the beginning of the era of error analysis. See, e.g., [W65] where in the introduction the author addressed this issue and noted that “*I felt that it is no longer practical to cover almost all known methods and to give an error analyses for them and decided to include mainly those methods of which I had extensive practical experience.*”

**2.3. Special classes of matrices and avoidance of pivoting.** Nevertheless, there are classes of matrices for which it is advantageous not to pivot! For example, for *totally positive*<sup>2</sup> matrices the exact triangular factors have only positive entries. De Boor and Pinkus pointed out in [DBP77] that if the entries of the computed factors  $\widehat{L}$  and  $\widehat{U}$  remain nonnegative, then the quantity  $g_n$  in (2.7) is of the order of unity, so that the componentwise backward and residual errors (2.4) and (2.6) are pleasantly small, and moreover,  $(R + \Delta R)\widehat{a} = f$  where

$$(2.8) \quad |\Delta R| \leq 3\gamma_n |R|.$$

**2.4. Vandermonde matrices again.** The above results give a theoretical justification for the numerical data in Tables 1 and 2. Indeed, the partial pivoting technique determines a permutation matrix  $P$ , such that at each elimination step the pivot elements  $d_k$  in

$$PR_1 = LU = \begin{bmatrix} 1 & & 0 \\ & \ddots & \\ * & & 1 \end{bmatrix} \begin{bmatrix} d_1 & & * \\ & \ddots & \\ 0 & & d_n \end{bmatrix}$$

are as large as possible. Clearly, the determinant of the leading  $k \times k$  submatrix of  $R_1$  is equal to  $d_1 \cdot \dots \cdot d_k$ , so we can say that the objective of partial pivoting is to successively maximize the determinants of the leading submatrices. This observation, and the well-known formula

$$\det V(x_{1:k}) = \prod_{\substack{1 \leq j \leq k \\ i < j}} (x_j - x_i)$$

for the determinant of the  $k \times k$  Vandermonde matrix easily imply that the Leja ordering (1.2) mimics the partial pivoting ordering of the rows of  $V(x)$ , see [Hig90]. So it is not surprising that the accuracy of GE with Leja ordering will be higher than with monotonic ordering, thus explaining the occurrence in Table 1.

Furthermore, it is nice to note that the condition  $0 < x_1 < \dots < x_n$  is well-known [GK50] to imply total positivity for the corresponding Vandermonde matrix, thus explaining the occurrence in Table 2.

*In brief, the well-known techniques stabilizing the structure-ignoring GE procedure can be efficiently implemented for the special Vandermonde structure as a*

---

<sup>2</sup>Totally positive matrices are defined as those for which the determinant of every submatrix is positive, see the classic monograph [GK50] for the definition and for several applications.

certain manipulation on the parameters defining such matrices. In the rest of the paper we show how this can be done for the other classes of structured matrices as well, and for fast algorithms exploiting their patterns of structure.

Before addressing structured matrices in detail we shall briefly describe next one more pivoting technique that we shall use below.

**2.5. Hermitian matrices and diagonal pivoting.** Matrices appearing in applications are often Hermitian, and exploiting this fact allows one to reduce the storage requirement by the factor 2, and since the symmetry is preserved during the elimination [see, e.g., (2.10) below], we can perform twice as less computations. It is therefore desirable not to lose symmetry under pivoting operations, so to perform symmetric permutations of rows and columns: In this section we recall the Bunch-Kaufman diagonal pivoting technique

$$(2.9) \quad R \leftarrow P^T R P.$$

Clearly, the main diagonal of  $P^T R P$  is a rearrangement of the main diagonal of  $R$ , so the reordering technique that brings the maximal magnitude element on the main diagonal to the pivotal position is called *symmetric pivoting*. Symmetric pivoting technique is sometimes used with positive definite matrices [GVL96], for which it is equivalent to complete pivoting. However, this method breaks down with indefinite matrices in the case when the whole main diagonal is zero. Moreover, even if the main diagonal will never become zero during elimination, symmetric pivoting in general cannot provide sufficiently large pivots, so this method is not backward stable for indefinite matrices.

The remedy [perhaps first suggested by W.M.Kahan] is to sometimes perform block elimination steps with appropriate  $2 \times 2$  block pivots. Symmetric block Gaussian elimination is based on recursive applying the well-known block Schur complementation formula for  $R_1 = \begin{bmatrix} R_{11} & R_{21}^* \\ R_{21} & R_{22} \end{bmatrix}$ :

$$(2.10) \quad R_1 = \begin{bmatrix} I & 0 \\ R_{21} \cdot R_{11}^{-1} & I \end{bmatrix} \cdot \begin{bmatrix} R_{11} & 0 \\ 0 & R_{22} - R_{21} \cdot R_{11}^{-1} \cdot R_{21}^* \end{bmatrix} \cdot \begin{bmatrix} I & R_{11}^{-1} \cdot R_{21}^* \\ 0 & I \end{bmatrix}.$$

The output of the algorithm is the block triangular factorization,

$$(2.11) \quad R_1 = LDL^*,$$

with a lower triangular factor  $L$  storing the columns of the form  $\begin{bmatrix} I \\ R_{21} R_{11}^{-1} \end{bmatrix}$ , and a block diagonal factor  $D$  storing the blocks of the form  $R_{11}$ . The task is to specify a recipe to choose an appropriate symmetric permutation (2.9) [providing the *block pivot*  $R_{11}$ ] to numerically stabilize the algorithm. For example, the LAPACK [LAPACK] package, renowned for the numerical reliability of the algorithms employed uses the *Bunch-Kaufman pivoting* technique. At each step the Bunch-Kaufman algorithm scans only two columns of the matrix  $R_1$  to determine the size  $m$  ( i.e., 1 or 2 ) of  $R_{11}$  and the permutation matrix  $P_1$  in

(2.10). To understand the Bunch-Kaufman technique it helps to consider the ma-

$$\text{trix } R_1 = \begin{bmatrix} r_{11} & \cdots & r_{t1}^* & \cdots & \cdots & \cdots \\ \vdots & & \vdots & & & \\ r_{t1} & \cdots & r_{tt} & \cdots & \sigma^* & \cdots \\ \vdots & & \vdots & & & \\ \vdots & & \sigma & & & \\ \vdots & & \vdots & & & \\ r_{11} \text{ or } r_{tt} \text{ or } & \begin{bmatrix} r_{11} & r_{t1}^* \\ r_{t1} & r_{tt} \end{bmatrix} \end{bmatrix}, \text{ and to note that the pivot } R_{11} \text{ is either}$$

### The Bunch-Kaufman algorithm [BK77]

```

 $\alpha = (1 + \sqrt{17})/8$ 
 $\lambda = |r_{t,1}| = \max\{|r_{2,1}|, \dots, |r_{n,1}|\}$  %This step determines t
if  $\lambda \neq 0$ 
  if  $|r_{1,1}| \geq \alpha\lambda$ 
     $m = 1; P = I$ 
  else
     $\sigma = |r_{p,t}| = \max\{|r_{1,t}|, \dots, |r_{t-1,t}|, |r_{t+1,t}|, \dots, |r_{n,t}|\}$ 
    if  $\sigma|r_{1,1}| \geq \alpha\lambda^2$ 
       $s = 1; P = I$ 
    else if  $|r_{t,t}| \geq \alpha\sigma$ 
       $m = 1 \text{ and choose } P \text{ so } (P \cdot R_1 \cdot P^*)_{1,1} = r_{t,t}$ 
    else
       $m = 2 \text{ and choose } P \text{ so } (P \cdot R_1 \cdot P^*)_{2,1} = r_{t,p}$ 
    end
  end
end

```

The value of the constant  $\alpha = (1 + \sqrt{17})/8$  is determined to bound the element growth [BK77], [B71]; this method was recently shown to be backward stable in [Hig95].

### 3. Exploiting structure: Fast GE with fast partial pivoting (Fast GEPP)

**3.1. Basic examples of structure.** In many applications in mathematics and engineering the underlying physical properties of the problem introduce various patterns of structure into arising matrices, some of these structures are shown in Table 3. It turns out that many problems involving these and more general patterns of structure can be nicely approached by using the unifying concept of *displacement*. This concept was introduced in [FMKL79] [KKM79] first in connection with Toeplitz matrices, and then the range of application of this approach was significantly extended in [HR84] to attack the other patterns of structure including those in Table 3. We next briefly recall these now well-known observations and definitions.

**3.2. Displacement structure.** For each of the above patterns of structure it is possible to choose a pair of auxiliary matrices  $\{F, A\}$  to define a *displacement operator*  $\nabla_{\{F,A\}}(\cdot) : \mathbf{C}^{n \times n} \rightarrow \mathbf{C}^{n \times n}$  of the form

$$(3.1) \quad \nabla_{\{F,A\}}(R) = FR - RA.$$

Toeplitz, $T = [ t_{i-j} ]$ $\begin{bmatrix} t_0 & t_{-1} & \cdots & \cdots & t_{-n+1} \\ t_1 & t_0 & t_{-1} & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & t_0 & t_{-1} \\ t_{n-1} & \cdots & \cdots & t_1 & t_0 \end{bmatrix}$	Hankel, $H = [ h_{i+j-2} ]$ . $\begin{bmatrix} h_0 & h_1 & h_3 & \cdots & h_{n-1} \\ h_1 & h_2 & & \ddots & h_n \\ h_2 & & \ddots & \ddots & \vdots \\ \vdots & & h_{n-1} & h_n & h_{2n-3} \\ h_{n-1} & h_n & \cdots & h_{2n-3} & h_{2n-2} \end{bmatrix}$
Vandermonde, $V = [ x_i^{j-1} ]$ $\begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^{n-1} \end{bmatrix}$	Cauchy, $C = [ \frac{1}{x_i - y_j} ]$ $\begin{bmatrix} \frac{1}{x_1 - y_1} & \cdots & \frac{1}{x_1 - y_n} \\ \vdots & & \vdots \\ \frac{1}{x_n - y_1} & \cdots & \frac{1}{x_n - y_n} \end{bmatrix}$
Polynomial Vandermonde, $V_P = [ P_{j-1}(x_i) ]$ $\begin{bmatrix} P_0(x_1) & \cdots & P_{n-1}(x_1) \\ \vdots & & \vdots \\ P_0(x_n) & \cdots & P_{n-1}(x_n) \end{bmatrix}$ where $\deg P_k(x) = k$	Pick, $P = \begin{bmatrix} \frac{\varphi_i^* \cdot J \cdot \varphi_j}{z_i + z_j^*} \\ \frac{\varphi_1 \cdot J \cdot \varphi_1^*}{z_1 + z_1^*} & \cdots & \frac{\varphi_1 \cdot J \cdot \varphi_n^*}{z_1 + z_n^*} \\ \vdots & & \vdots \\ \frac{\varphi_n \cdot J \cdot \varphi_1^*}{z_n + z_1^*} & \cdots & \frac{\varphi_n \cdot J \cdot \varphi_n^*}{z_n + z_n^*} \end{bmatrix}$ where $\varphi_k \in \mathbf{C}^{1 \times \alpha}$ , $J \in \mathbf{C}^{\alpha \times \alpha}$

TABLE 3. Some examples of matrices with structure.

The point is that for each class of matrices with a particular pattern of structure, such as Toeplitz, Vandermonde, etc., the pair  $\{F, A\}$  can be chosen to map matrices from this class to low rank matrices. Moreover, the number

$$\alpha_{\{F, A\}}(R) = \operatorname{rank}(FR - RA)$$

is called a  $\nabla_{\{F, A\}}$ -displacement rank.

For example, we started the paper with an example involving a Vandermonde matrix,  $V(x)$ , for this pattern of structure we can choose  $F = D_x = \operatorname{diag}(x_1, \dots, x_n)$ ,  $A = Z_1$ , the cyclic lower shift matrix having ones on the first subdiagonal and in the (1,n) position, and zeros elsewhere. Then we have

$$(3.2) \quad \nabla_{\{D_x, Z_1\}}(V(x)) = D_x V(x) - V(x)Z_1 = \begin{bmatrix} x_1^n - 1 \\ \vdots \\ x_n^n - 1 \end{bmatrix} \begin{bmatrix} 0 & \cdots & 0 & 1 \end{bmatrix}.$$

Briefly, ordinary Vandermonde matrices have  $\nabla_{\{D_x, Z_1\}}$ -displacement rank one. If the  $\nabla_{\{D_x, Z_1\}}$ -displacement rank of  $R$  is bigger than one, but still much smaller than the size of the matrix, then it is natural to refer to such  $R$  as a *Vandermonde-like* matrix. Such Vandermonde-like matrices appear in several applications, e.g., in inverse spectral problems for matrix polynomials [GLR82].

Similar justifications can be given for the names of the other “-like” matrices, listed in Table 4. In this survey we will be especially interested in Cauchy-like matrices, defined via the displacement equation  $\nabla_{\{D_x, D_y\}}(R) = D_x R - RD_y$ , so let us first observe that Pick matrices,  $P = \left[ \frac{\varphi_i^* J \varphi_j^*}{z_i + z_j^*} \right]$ , defined in Table 3 are

Name	Choice of $F$	Choice of $A$
Toeplitz-like [FMKL79], [KKM79], [AG90], [GO92], [GO94c]	$\bar{F} = Z_\gamma$	$A = Z_\delta$
Hankel-like	$\bar{F} = Z_\gamma$	$A = Z_\delta^T$
Toeplitz-plus-Hankel-like [GKO95] <sup>3</sup>	$\bar{F} = Y_{\gamma_1, \delta_1}$	$A = Y_{\gamma_2, \delta_2}$
Cauchy-like [HR84]	$\bar{F} = \text{diag}(c_1, \dots, c_n)$	$A = \text{diag}(d_1, \dots, d_n)$
Vandermonde-like [HR84], [GO94c] <sup>4</sup>	$\bar{F} = \text{diag}(\frac{1}{x_1}, \dots, \frac{1}{x_n})$	$A = Z_\gamma$
Chebyshev–Vandermonde-like [KO95]	$\bar{F} = \text{diag}(x_1, \dots, x_n)$	$A = Y_{\gamma, \delta}$
three-term–Vandermonde-like [KO97a]	$\bar{F} = \text{diag}(x_1, \dots, x_n)$	Comrade matrix
polynomial–Vandermonde-like [KO97a]	$\bar{F} = \text{diag}(x_1, \dots, x_n)$	Confederate matrix
Chebyshev–Hankel-like [KO97b]	Comrade matrix	Comrade matrix
Polynomial–Hankel-like [KO97b]	Confederate matrix	Confederate matrix

TABLE 4. Basic examples of displacement structure with choices for  $F$  and  $A$  in (3.1). Here  $Z_\gamma$  is the lower shift  $\gamma$ -circulant matrix.

$Y_{\gamma, \delta} = Z_0 + Z_0^T + \gamma e_1 e_1^T + \delta e_n e_n^T$ , and the comrade and confederate matrices are defined as in [MB79] (we recall these definitions in Sec. 7.7.4 below).

Cauchy-like:

$$\nabla_{\{D_z, -D_z^*\}} = D_z P + P D_z^* = \begin{bmatrix} \varphi_1 \\ \vdots \\ \varphi_n \end{bmatrix} J \begin{bmatrix} \varphi_1^* & \cdots & \varphi_n^* \end{bmatrix}.$$

Indeed, their  $\nabla_{\{D_z, -D_z^*\}}$ -displacement rank is equal to  $\alpha$ , the length of the involved row vectors  $\varphi_k$ . As we shall recall in Sec. 6.1, the displacement rank  $\alpha$  of a Pick matrix is the size of the associated  $\alpha \times \alpha$  rational matrix function  $W(z)$ , and it is usually much smaller than the size of the Pick matrix  $P$  itself, which will be equal to the number of tangential interpolation conditions we wish  $W(z)$  to satisfy.

**3.3. Generators and design of fast algorithms.** Returning to the fact that all matrices in Table 3 are defined by a small number  $O(n)$  of parameters, one could

<sup>3</sup>The displacement structure approach has been used, e.g., in [HJR88], [GK89], [SLAK93] to study the ordinary Toeplitz-plus-Hankel matrices. In [GKO95] algorithms for general Toeplitz-plus-Hankel-like matrices have been designed.

<sup>4</sup>The fact that Vandermonde matrices has displacement rank one was first observed in [HR84], but here we follow [GO94c] and use a slightly different form (3.2) for displacement equation [i.e., with  $Z_1$  instead of  $Z_0$ , because this form is more convenient for transformations of Vandermonde-like matrices to Toeplitz-like or Cauchy-like matrices defined next]. See [GO94c] or Sec. 7.1 below for details.

comment that the dependence of the entries upon these parameters is quite different for each particular pattern. The importance of the displacement rank concept is that it unifies such a dependence, as described next. Let  $\alpha \stackrel{\Delta}{=} \text{rank} \nabla_{\{F,A\}}(R)$ , then one can factor [nonuniquely]

$$(3.3) \quad \nabla_{\{F,A\}}(R) = FR - RA = GB^T,$$

where both rectangular matrices on the right-hand side of (3.3) have only  $\alpha$  columns each:  $G, B \in \mathbf{C}^{n \times \alpha}$ . The pair  $\{G, B\}$  is called a  $\nabla_{\{F,A\}}$ -generator of  $R$ . Thus we see that the displacement rank measures the complexity of  $R$ , because all its  $n^2$  entries are described by a smaller number  $2\alpha n$  entries of its generator  $\{G, B\}$ . This gives us a clue on how to exploit the structure to devise a certain fast algorithm we may need:

*translate operations on entries of  $R$  to the language of operations on its generator  $\{G, B\}$ .*

This approach can be used to efficiently obtain fast implementations for various algorithms, and in the next subsection it is used to speed-up the Gaussian elimination procedure.

**3.4. Exploiting the displacement structure to speed-up the block GE procedure.** As is well-known, the classical Schur algorithm [S17] for checking if the analytic function is bounded by unity in the interior of unit disk can be seen as a *fast*  $O(n^2)$  method to compute Cholesky factorization  $T = LL^*$  for a certain positive definite Toeplitz matrix  $T$ . See, e.g., [K86], [K87] and Ch.1 for details. Two years after [S17], Nevanlinna [N19] applied a clever modification of this algorithm to solve a closely related, now classical, scalar Nevanlinna-Pick interpolation problem, see, e.g., [A65] and the references therein. It is widely known as well that the classical Nevanlinna algorithm can be seen as a *fast*  $O(n^2)$  method to compute the Cholesky factorization for the associated Pick matrix,  $P = \begin{bmatrix} 1-f_i f_j^* \\ z_i + z_j^* \end{bmatrix}$  [recall that the Nevanlinna-Pick problem is solvable if and only if the Pick matrix is positive definite, which ensures the existence of the Cholesky factorization  $P = L L^*$ ].

Now note that the usual Toeplitz or Pick matrices are just two particular examples of matrices with displacement structure [in both cases the corresponding displacement rank is  $\leq 2$ ], and recall that the triangular factorization can be done in  $O(n^2)$  operations not only for these two particular patterns of structure, but for all kinds of displacement structure listed in Table 4. The crucial fact [explicitly first noted perhaps by M.Morf, e.g., in [M80]] that makes the speed-up possible can be vaguely formulated as follows:

*the displacement structure of a matrix is inherited under Schur complementation.*

Hence the successive Schur complementation (2.1) of the Gaussian elimination procedure

$$(3.4) \quad \begin{array}{ccccc} R_1 & \rightarrow & R_2 & \rightarrow & \dots \rightarrow & R_n \\ \downarrow & & \downarrow & & & \downarrow \\ \{l_1, u_1\} & & \{l_2, u_2\} & & \dots & \{l_n, u_n\} \end{array}$$

can be translated to the language of operations on generators.

$$(3.5) \quad \begin{array}{ccccccc} \{G_1, B_1\} & \rightarrow & \{G_2, B_2\} & \rightarrow & \dots & \rightarrow & \{G_n, B_n\} \\ \downarrow & & \downarrow & & & & \downarrow \\ \{l_1, u_1\} & & \{l_2, u_2\} & & \dots & & \{l_n, u_n\} \end{array}$$

Algorithms implementing in  $O(n^2)$  operations [or faster] the generator recursion (3.5) have been called *fast Cholesky* [or, in the non-symmetric case, *fast GE*] algorithms, but now they are more often referred to as *Schur-type* and *generalized Schur algorithms*. It is perhaps impossible to list all relevant connections. the analysis of algorithmic issues (first for Toeplitz-like matrices) starts with a breakthrough work of Martin Morf in [M74], [M80], [D82] followed by, e.g., [LAK84], [LAK86], [GKKL87], [CKLA87], [C89], [KS92], [S92], [GO94b], [GKO95], [KO95], [BKO95b], [KS95], [LA96], [KO97a], [KO97b] among others. Any of these forms as well as of other possible extensions can be used for our purposes. it will be convenient to use here the following variant of a generator recursion admitting a simple one-line-proof [see Sec. 4.4, 4.5 for an interpolation meaning of this result].

**LEMMA 3.1.** ([GO93], [GO94b], [GKO95]) *Let matrix  $R_1$  be partitioned  $R_1 = \begin{bmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{bmatrix}$ , where we assume the upper left  $m \times m$  block  $R_{11}$  to be nonsingular. If  $R_1$  satisfies the displacement equation*

$$(3.6) \quad \nabla_{\{F_1, A_1\}}(R_1) = \begin{bmatrix} F_{11} & 0 \\ * & F_2 \end{bmatrix} \cdot R_1 - R_1 \cdot \begin{bmatrix} A_{11} & * \\ 0 & A_2 \end{bmatrix} = G_1 \cdot B_1,$$

where  $G_1 \in \mathbf{C}^{n \times \alpha}$ ,  $B_1 \in \mathbf{C}^{\alpha \times n}$ . Then the Schur complement  $R_2 = R_{22} - R_{21}R_{11}^{-1}R_{12}$  satisfies the displacement equation

$$(3.7) \quad F_2 \cdot R_2 - R_2 \cdot A_2 = G_2 \cdot B_2,$$

with

$$(3.8) \quad \begin{bmatrix} 0 \\ G_2 \end{bmatrix} = G_1 - \begin{bmatrix} I_m \\ R_{21}R_{11}^{-1} \end{bmatrix} \cdot G_{11}, \quad \begin{bmatrix} 0 & B_2 \end{bmatrix} = B_1 - B_{11} \cdot \begin{bmatrix} I_m & R_{11}^{-1}R_{12} \end{bmatrix}.$$

where  $G_{11}$  and  $B_{11}$  are the first  $m$  rows of  $G_1$  and the first  $m$  columns of  $B_1$ , respectively.

**Proof.** For your pleasure: from (3.6) and the standard Schur complementation formula,

$$R_1 = \begin{bmatrix} I_m & 0 \\ R_{21}R_{11}^{-1} & I_{n-m} \end{bmatrix} \cdot \begin{bmatrix} R_{11} & 0 \\ 0 & R_2 \end{bmatrix} \cdot \begin{bmatrix} I_m & R_{11}^{-1}R_{12} \\ 0 & I_{n-m} \end{bmatrix}.$$

it follows that

$$\begin{aligned} & \begin{bmatrix} F_{11} & 0 \\ * & F_2 \end{bmatrix} \cdot \begin{bmatrix} R_{11} & 0 \\ 0 & R_2 \end{bmatrix} - \begin{bmatrix} R_{11} & 0 \\ 0 & R_2 \end{bmatrix} \cdot \begin{bmatrix} A_{11} & * \\ 0 & A_2 \end{bmatrix} = \\ & \begin{bmatrix} I_m & 0 \\ -R_{21}R_{11}^{-1} & I_{n-m} \end{bmatrix} \cdot B_1 \cdot G_1 \cdot \begin{bmatrix} I_m & -R_{11}^{-1}R_{12} \\ 0 & I_{n-m} \end{bmatrix}. \end{aligned}$$

Equating the (2,2) block entries, we obtain (3.8).



**3.5. Fast implementation of partial pivoting.** Formulas (3.8) is one possible implementation of the generator recursion (3.5) allowing one to speed-up the Gaussian elimination procedure. Since pivoting is not incorporated at the moment, this would be a fast implementation of a numerically inaccurate algorithm, cf. with Sec. 2.1, 2.2. Applying partial pivoting requires replacing of the maximum magnitude entry in the first column of  $R_1$  to the (1,1) position using row interchange, or equivalently, by multiplication with the corresponding permutation matrix  $P_1$ , and then performing elimination :

$$(3.9) \quad P_1 \cdot R_1 = \begin{bmatrix} 1 & 0 \\ \frac{1}{d_1}l_1 & I \end{bmatrix} \cdot \begin{bmatrix} d_1 & u_1 \\ 0 & R_2 \end{bmatrix}.$$

Since we assume that  $R_1$  satisfies (3.6), its permuted version,  $P_1 R_1$ , will satisfy

$$(3.10) \quad (P_1 F_1 P_1^T)(P_1 R_1) - (P_1 R_1)A_1 = (P_1 G_1)B_1^T.$$

Since we can run the generator recursion (3.8) only for lower triangular ( $P_1 F_1 P_1^T$ ), and since we do not have any a-priori restrictions on the permutation  $P_1$ , we have to assume  $F_1$  to be a diagonal matrix. Summarizing, a row interchange does not destroy the Cauchy-like, Vandermonde-like, Chebyshev–Vandermonde-like and polynomial–Vandermonde-like displacement structures. In this case pivoting can be incorporated, and it is equivalent to the update

$$F_1 \leftarrow P_1 \cdot F_1 \cdot P_1^T, \quad G_1 \leftarrow P_1 \cdot G_1,$$

leading to the computational procedure, formulated next.

Fast GEPP for a structured matrix with diagonal  $F_1$

**INPUT:** A  $\nabla_{\{F_1, A_1\}}$ -generator  $\{G_1, B_1\}$  in

$$(3.11) \quad F_1 R_1 - R_1 A_1 = G_1 B_1$$

with diagonal  $F_1$ .

**OUTPUT:** The permuted triangular factorization  $PR = LU$ .

**1:** First recover from the generator the first column of  $R_1$ . The computations for doing this depend on the form of the matrices  $F_1$  and  $A_1$  in displacement equation (3.11), but for each kind of structure it is immediate to write down the corresponding formula, see, e.g., [GKO95], [KO95], [KO97a]. For example, for Cauchy-like matrices in  $D_x R_1 - R_1 D_y = G_1 B_1$ , we have  $r_{ij} = \frac{g_i \cdot b_j}{x_i - y_j}$ , where  $g_i$  is the  $i$ -th row of  $G_1$  and  $b_j$  is the  $j$ -th column of  $B_1$ .

**2:** Next determine the position, say (k,1), of the entry with maximal magnitude in the recovered first column. Let  $P_1$  be a permutation of the 1-st and the  $k$ -th entries. Interchange the 1-st and the  $k$ -th diagonal entries of  $F_1$  in (1.2); interchange the 1-st and  $k$ -th rows in the matrix  $G_1$  in (1.2).

**3:** Then recover from the generator the first row of  $P_1 \cdot R_1 = \begin{bmatrix} d_1 & u_1 \\ l_1 & R_{22}^{(1)} \end{bmatrix}$ .

Now store  $\begin{bmatrix} 1 \\ \frac{1}{d_1}l_1 \end{bmatrix}$  as the first column of  $L$  and  $[d_1 \ u_1]$  as the first row of  $U$  in the LU factorization of the permuted matrix,  $P_1 \cdot R_1$  in (3.9).

**4:** Next compute by (3.8) a generator of the Schur complement  $R_2 = R_{22}^{(1)} - \frac{1}{d_1}l_1 u_1$  of  $P_1 \cdot R_1$  in (3.9).

**5:** Proceed recursively with  $R_2$  which is now represented its generator  $\{G_2, B_2\}$  to finally obtain the factorization  $R_1 = P \cdot L \cdot U$ , where  $P = P_1 \cdot P_2 \cdot \dots \cdot P_{n-1}$  with  $P_k$  being the permutation used at the  $k$ -th step of the recursion.

We refer to [GO93], [GO94b], [GKO95], [KO97a], [KO97a] for many computed examples. some of these examples will be reproduced below.

**3.6. A problem: non-optimality for Hermitian matrices.** It has been just shown that GEPP can be efficiently implemented to rapidly factorize in  $O(n^2)$  operations Vandermonde-like, polynomial-Vandermonde-like and Cauchy-like matrices. In many applications however, Cauchy-like matrices are usually *Hermitian* [e.g., Pick matrices in Table 3], satisfying

$$(3.12) \quad \nabla_A(R) = A^*R + RA = GJG^*.$$

or

$$(3.13) \quad \nabla_F(R) = R - FRF^* = GJG^*.$$

Of course, partial pivoting destroys the symmetry of such matrices. so the above fast GEPP algorithm is not optimal for the Hermitian case.

To fully exploit the Hermitian structure of non-structured matrices one usually uses diagonal pivoting, see, e.g., [GV96]; this allows to obtain twice as efficient algorithm in terms of both speed and memory. In Sec. 5 we will design its fast implementation for Hermitian Cauchy-like matrices. However, before doing so we describe next an important application of partial pivoting to rational interpolation problems.

#### 4. Partial pivoting and accurate solution of tangential interpolation problems

**4.1. State-space method.** In the early 1960's R.E.Kalman introduced a state-space method to study linear time-invariant dynamical systems. This method is based on a representation of the transfer function for such systems, i.e., a rational  $m \times m$  matrix function  $W(z) = \begin{bmatrix} p_{ij}(z) \\ q_{ij}(z) \end{bmatrix}$ , in the form called a *realization*.

$$(4.1) \quad W(z) = D + C(zI - A)^{-1}B,$$

with the matrices  $D \in \mathbf{C}^{\alpha \times \alpha}$ ,  $C \in \mathbf{C}^{\alpha \times n}$ ,  $A \in \mathbf{C}^{n \times n}$ ,  $B \in \mathbf{C}^{n \times \alpha}$  of appropriate sizes. The realization is called *minimal*, if the size of  $A$  is the smallest possible.

Although the original interest in the state-space method was for specific engineering applications [e.g., the linear quadratic regulator problem and the Kalman filter], the state-space method turned out to be fundamental, leading to new insights in many other directions. Various engineering and mathematical problems have been addressed in this way; however, in this section we restrict ourselves only to several rational matrix interpolation problems, such as those of Lagrange-Sylvester, Schur, Nevanlinna-Pick, Caratheodory-Fejer, Nehari and Nehari-Takagi. We refer to a monograph [BGR90] for a systematic treatment of all these interpolation problems, as well as for their use to solve various engineering problems [sensitivity minimization, model reduction, robust stabilization], and for a wide list of references and historical remarks. Given such a vast literature on interpolation, we have to explain the need for the further analysis, especially in the simplest frameworks taken in this section. The point is that although there are a number

of explicit solutions for a variety of such problems, the numerical properties of the corresponding computational schemes were totally ignored. As a result, there is a clear gap in practically reliable software tools for solving even simplest rational interpolation problems. In this section we show how the state-space method can provide not only mathematical descriptions, but also accurate and fast practical algorithms to compute solutions.

The attractive property of the representation (4.1) is that it can usually reduce a certain analytic problem involving  $W(z)$  to just a linear algebra problem, i.e. to just standard manipulations on finite matrices, such as inversion, triangular factorization, solving the associated linear system, etc. Such a reduction seems to be beneficial for practical purposes, because it allows us to compute solutions for a variety of applied problems by using available standard linear algebra software tools. However, there are two difficulties. First, the size of matrices can be quite large, so computing the solution via general purpose algorithms may require substantial storage and time. As we shall see in a moment, particular interpolation problems often introduce a displacement structure into the corresponding matrices. This structure, of course, can be exploited to speed-up the computation, however, without any stabilizing technique such algorithms would be impractical, leading to the rapid computing of a numerically inaccurate solution. As we shall show the fast GEPP algorithm of Sec. 3.5, and is applicable, and it provides not only fast, but also more accurate, solutions.

To sum up, fast implementations of various pivoting techniques have a variety of important applications improving the accuracy of many algorithms for rational matrix interpolation and for solving several engineering problems, listed, e.g., in [BGR90].

**4.2. Simplest example: homogeneous tangential interpolation problem and Cauchy-like matrices.** In fact, in a variety of interpolation problems a rational matrix function  $W(z)$  is often not given by the four matrices  $\{A, B, C, D\}$  in (4.1), but by some other set of matrices, similar to the situation in the next example.

---

The simplest homogeneous tangential interpolation problem

---

$$\begin{array}{ll} n \text{ distinct points} & \{x_1, \dots, x_n\}, \\ \text{Given: } & \{ \varphi_1, \dots, \varphi_n \}, \\ n \text{ nonzero } 1 \times \alpha \text{ row vectors} & \\ n \text{ distinct points} & \{y_1, \dots, y_n\}, \\ n \text{ nonzero } \alpha \times 1 \text{ column vectors} & \{ \psi_1, \dots, \psi_n \}. \end{array}$$

[Let us assume  $\{x_k, y_k\}$  to be  $2n$  distinct complex numbers.]

**Construct:** A rational  $\alpha \times \alpha$  matrix function  $W(z)$  having the identity value at infinity, and such that

- *Left tangential interpolation condition:*  $\det W(z)$  has a simple zero at  $x_k$ , and

$$(4.2) \quad \varphi_k W(x_k) = 0.$$

- *Right tangential interpolation condition:*  $\det W(z)^{-1}$  has a simple zero at  $y_k$ , and

$$(4.3) \quad W(y_k)^{-1} \psi_k = 0.$$

**Solvability condition:** The unique [in this simples case] solution  $R$  of the displacement equation

$$(4.4) \quad \boxed{RA_\pi - A_\zeta R = B_\zeta C_\pi}$$

should be invertible, where we define

$$C_\pi = [\psi_1 \dots \psi_n], \quad A_\pi = \text{diag} [y_1 \dots y_n].$$

$$A_\zeta = \text{diag} [x_1 \dots x_n], \quad B_\zeta = [\varphi_1^T \dots \varphi_n^T]^T.$$

**Solution:** , as given in [BGR90]:

$$(4.5) \quad \boxed{W(z) = I + C_\pi(zI - A_\pi)^{-1}R^{-1}B_\zeta.}$$

So, the eigenvalues of  $A_\pi$  are the poles, and the eigenvalues of  $A_\zeta$  are the zeros of  $W(z)$  [explaining the choice for the subscripts to denote these matrices]. We see that in order to couple the *null data*  $\{A_\zeta, B_\zeta\}$  and the *pole data*  $\{C_\pi, A_\pi\}$  we need to bring into the consideration the displacement equation (4.4), whose solution  $R$  has been called a *null-pole coupling matrix* in [BGR90]. Now notice that the entries of  $R$  are not a part of the given data, and in this particular problem  $R$  is defined implicitly, by its generator  $\{B_\zeta, C_\pi\}$ .

**4.3. Homogeneous interpolation and other classes of matrices with displacement structure.** The above example with *diagonal*  $A_\pi$  and  $A_\zeta$  [thus giving rise to a Cauchy-like matrix  $R$  in (4.5)] is simplest but by no means separate. In fact, the monograph [BGR90] is a manifestation of the application of explicit formula (4.5) to solve a wide variety of interpolation problems. including those listed at the beginning of Sec. 4.1. In all these problems a structured matrix  $R$  is also defined implicitly, by the two pairs of matrices, the “pole pair”  $\{C_\pi, A_\pi\}$  for a minimal realization

$$(4.6) \quad W(z) = I + C_\pi(zI - A_\pi)^{-1}B_\pi.$$

[with  $B_\pi$  unknown], and another, “null pair”  $\{A_\zeta, B_\zeta\}$  for a minimal realization

$$W(z)^{-1} = I + C_\zeta(zI - A_\zeta)^{-1}B_\zeta,$$

[now with  $C_\zeta$  unknown]. As above, in order to recover a rational matrix function  $W(z)$  from the null data  $\{A_\zeta, B_\zeta\}$  and the pole data  $\{C_\pi, A_\pi\}$ , we need a *null-pole coupling matrix*  $R$  defined as a solution of the displacement equation

$$(4.7) \quad RA_\pi - A_\zeta R = B_\zeta C_\pi$$

with arbitrary  $\{A_\pi, A_\zeta\}$ , so all special structures of Table 4 are captured. The now well-known result of [BGR90] states that for a given  $W(z)$  [i.e., given by the two pairs  $\{C_\pi, A_\pi\}$  and  $\{A_\zeta, B_\zeta\}\}$  there is a unique invertible solution to (4.7) satisfying

$$(4.8) \quad W(z) = I + C_\pi(zI - A_\pi)^{-1}R^{-1}B_\zeta$$

[if there are many solutions  $R$  to (4.7), then each of them defines a different  $W(z)$  via (4.8)]. The above collection of matrices

$$(4.9) \quad \tau = \{C_\pi, A_\pi, A_\zeta, B_\zeta, R\}$$

has been called called a *global left null-pole triple*<sup>5</sup> for  $W(z)$ .

---

<sup>5</sup>The number of matrices in the triple shows a tendency to grow over the years. Indeed, in [GLR82] triple still consisted of three matrices.

However, this seems to be a typical example of a situation when there is a complete mathematical description of a solution for an important class of problem, but there are no numerical algorithms to compute these solutions with good accuracy and rapidly. Indeed, the representation (4.7) is not particularly convenient for practical purposes [if we need to further evaluate  $W(z)$ ], because (4.8) explicitly involves the inverse of  $R$ .

We next follow [GO93] [GO94c] to present two fast and accurate algorithms to compute more convenient representations for  $W(z)$ . Basically, there are two methods to replace (4.8) by more appropriate forms. First, we can solve  $\alpha$  linear systems of equations,

$$(4.10) \quad RB_\pi = B_\zeta,$$

to replace (4.8) by (4.1). This approach does not require any further explanations, and we only refer to many computed examples in [GO94c], where  $R$  had a Hermitian partially reconstructible Cauchy-like structure [imposed by the Nehari problem, to be described later in Sec. 6.5]. In that context (4.10) was solved via the slow GECP [complete pivoting], and via the fast algorithms for solving Cauchy-like linear equations, including those described in Sec. 3.5 [based on partial pivoting] and those to be described later in Sec. 5.3, 6.3 [based on diagonal pivoting]. The conclusions were that without pivoting such fast methods are inaccurate, and the slow GECP provides much better results. However there is no need to “pay by the speed for accuracy”, and a fast implementation of partial and diagonal pivoting techniques was shown to enhance the numerical behavior of these fast methods, so they provide the same accuracy as slow GECP, while computing the solution much faster.

We next describe how to obtain one more convenient representation for  $W(z)$ .

**4.4. Factorization of rational matrix functions.** Another more convenient than (4.8) representation of  $W(z)$  would be its cascade decomposition,

$$(4.11) \quad W(z) = \Theta_1(z) \cdot \dots \cdot \Theta_n(z) \quad \text{with} \quad \Theta_k(z) = I + c_k \frac{1}{z - x_1} \frac{1}{d_k} b_k,$$

i.e.,  $\Theta_k(z)$  are the first-order factors, having just one pole and one zero. Since  $c_k$  is just a  $\alpha \times 1$  column,  $d_k$  is a scalar, and  $b_k$  is just a  $1 \times \alpha$  row, the representation (4.11) is very attractive, because it allows us to evaluate  $W(z)$  in only  $O(\alpha n)$  operations. This motivates us to look for a numerically reliable fast algorithm to compute a factorization for rational matrix functions. As we shall see in a moment, such a factorization of  $W(z)$  is just a Gaussian elimination on its null-pole coupling matrix  $R$  [this fact is known, see, e.g., [S86]], and the point is that fast implementations of several pivoting techniques are to be employed to achieve accuracy.

To present an algorithm, we need to recall several widely known results. There is a large number of theorems, e.g., in [P60], [S86], [BGKV80], [GVKDM83], [AD86], [AG88], [LAK92], [GO93], [SKLAC94], [GO94b] on factorization of rational matrix functions. In particular, the following statement is well-known.

THEOREM 4.1. [ [BGKV80]<sup>6</sup>, [BGK79], Theorem 4.8] *Let*

$$(4.12) \quad W_1(z) = I_p + C \cdot (zI_N - A)^{-1} \cdot B$$

---

<sup>6</sup>Interestingly, the paper [BGKV80] followed a curious situation, in which Bart-Gohberg-Kaashoek and P.Van Dooren reported this [independently obtained] result in two consecutive talks.

be a minimal realization. If matrices  $A \in \mathbf{C}^{N \times N}, B \in \mathbf{C}^{N \times p}, C \in \mathbf{C}^{p \times N}$  can be partitioned such that matrix  $A$  is upper triangular :

$$C = [ \begin{array}{cc} C_1 & C_2 \end{array} ], \quad A = [ \begin{array}{cc} A_1 & * \\ 0 & A_2 \end{array} ]. \quad B = [ \begin{array}{c} B_1 \\ B_2 \end{array} ].$$

and matrix  $A^\times = A - B \cdot C$  is lower triangular :

$$A^\times = [ \begin{array}{cc} A_1^\times & 0 \\ * & A_2^\times \end{array} ].$$

then the rational matrix function  $W_1(z)$  admits a minimal factorization  $W_1(z) = \Theta_1(z) \cdot W_2(z)$ , where

$$(4.13) \quad \Theta_1(z) = I_p + C_1 \cdot (zI_{N_1} - A_1)^{-1} \cdot B_1, \quad W_2(z) = I_p + C_2 \cdot (zI_{N_2} - A_2)^{-1} \cdot B_2.$$

Moreover, each minimal factorization of  $W_1(z)$  can be obtained in this way.

Here we have only to explain the name “minimal factorization” for  $W_1(z) = \Theta_1(z) \cdot W_2(z)$ . This simply means that there is no “zero-pole cancelation”, i.e., the size of  $A$  in (4.12) is equal to the sum of the sizes of  $A_1$  and  $A_2$  in (4.13).

The above theorem factorizes  $W_1(z)$  given in the form of realization (4.12). This result is not appropriate for our purpose, because in our applications  $W_1(z)$  will be given in the form (4.8), i.e., by its global left null-pole triple (4.9). The following theorem describes a variant of the above theorem in the desired form.

**THEOREM 4.2.** [ [GO93] [GO94c], Theorem 2.3 ] Let  $\tau_1 = (C_\pi, A_\pi, A_\zeta, R_1)$  be a global left null-pole triple for rational matrix function

$$W_1(z) = I + C_\pi(zI - A_\pi)^{-1} R_1^{-1} B_\zeta,$$

with identity value at infinity. Let matrices in  $\tau_1$  be partitioned as

$$(4.14) \quad \begin{aligned} C_\pi &= [ \begin{array}{cc} C_{\pi,1} & C_{\pi,2} \end{array} ], & A_\pi &= [ \begin{array}{cc} A_{\pi,1} & * \\ 0 & A_{\pi,2} \end{array} ], \\ A_\zeta &= [ \begin{array}{cc} A_{\zeta,1} & 0 \\ * & A_{\zeta,2} \end{array} ], & B_\zeta &= [ \begin{array}{c} B_{\zeta,1} \\ B_{\zeta,2} \end{array} ], \\ R_1 &= [ \begin{array}{cc} R_{11} & R_{12} \\ R_{21} & R_{22} \end{array} ]. \end{aligned}$$

If matrix  $R_{11}$  is invertible then  $W_1(z)$  admits a minimal factorization  $W_1(z) = \Theta_1(z) \cdot W_2(z)$ , where

$$(4.15) \quad \tau_\Theta = (C_{\pi,1}, A_{\pi,1}, A_{\zeta,1}, B_{\zeta,1}, R_{11})$$

is a global left null-pole triple for

$$(4.16) \quad \Theta_1(z) = I_p + C_{\pi,1} \cdot (zI_{N_1} - A_{\pi,1})^{-1} \cdot R_{11}^{-1} \cdot B_{\zeta,1},$$

and

$$(4.17) \quad \begin{aligned} \tau_2 &= (C_{\pi,2} - C_{\pi,1} \cdot R_{11}^{-1} \cdot R_{12}, \quad A_{\pi,2}, \quad A_{\zeta,2}, \\ &\quad B_{\zeta,2} - R_{21} \cdot R_{11}^{-1} \cdot B_{\zeta,1}, \quad R_{22} - R_{21} \cdot R_{11}^{-1} \cdot R_{12}) \end{aligned}$$

is a global left null-pole triple for

$$(4.18) \quad \begin{aligned} W_2(z) &= I_p + \\ &+ (C_{\pi,2} - C_{\pi,1} \cdot R_{11}^{-1} \cdot R_{12})(zI_{N_2} - A_{\pi,2})^{-1} \cdot (R_{22} - R_{21} \cdot R_{11}^{-1} \cdot R_{12})^{-1} \cdot (B_{\zeta,2} - R_{21} \cdot R_{11}^{-1} \cdot B_{\zeta,1}) \end{aligned}$$

**4.5. Computational aspects of theorems 4.1-4.2.** Here we address computational implications of theorem 4.2, and observe that it leads to a recursive algorithm for factorization of rational matrix functions, and that this algorithm in fact coincides with the simple recursion of Lemma 3.1. To clarify this, let us make the following three comments.

- For a given  $W_1(z)$ , its cascade decomposition (4.11) can be computed recursively:  $W_1(z) \rightarrow \{\Theta_1(z), W_2(z)\}$ , then  $W_2(z) \rightarrow \{\Theta_2(z), W_3(z)\}$ , etc. Formulas in (4.17) reduce this function recursion to an array algorithm,

$$(4.19) \quad \begin{array}{ccccccc} \tau_1 & \rightarrow & \tau_2 & \rightarrow & \dots & \rightarrow & \tau_{n-1} & \rightarrow & \tau_n \\ \downarrow & & \downarrow & & & & \downarrow & & \downarrow \\ \Theta_1(z) & & \Theta_2(z) & & \dots & & \Theta_{n-1}(z) & & \Theta_n(z) \end{array},$$

where one manipulates on just matrices  $\{B_\zeta, C_\pi\}$  in global left null-pole triples  $\tau_k$ .

- The factorization steps  $W_k(z) = \Theta_k(z) \cdot W_{k+1}(z)$  correspond to the step of Gaussian elimination on the null-pole coupling matrix  $R_1$ , e.g.:

$$\begin{array}{ccc} W_1(z) & \rightarrow & W_2(z) \\ \downarrow & & \downarrow \\ R_1 & & R_2 := R_{22} - R_{21}R_{11}^{-1}R_{12} \end{array}$$

To see this just note that the null-pole coupling matrix in (4.17) is the Schur complement  $R_2$ .

In fact, the this second comment is known, see [S86] [and earlier in Russian], however the author of [S86] was not interested in computational aspects, so his variant of a factorization theorem lead to a “superslow”  $O(n^4)$  algorithm.

Now we are ready to explain how theorem 4.2 leads to a fast  $O(n^2)$  factorization algorithm, and the explanation is based on its connection to Lemma 3.1. By exploring the definition (4.7) for the null-pole coupling matrices of the initial function  $W_1(z)$ , as well as of the quotient  $W_2(z)$ , specified by theorem 4.2, we have:

$$R_1 A_\pi - A_\zeta R_1 = B_\zeta C_\pi,$$

and

$$R_2 A_{\pi,2} - A_{\zeta,2} R_2 = (B_{\zeta,2} - R_{21} \cdot R_{11}^{-1} \cdot B_{\zeta,1})(C_{\pi,2} - C_{\pi,1} \cdot R_{11}^{-1} \cdot R_{12}),$$

where  $R_2 \triangleq (R_{22} - R_{21} \cdot R_{11}^{-1} \cdot R_{12})$  is the Schur complement. By comparing these formulas to the generator recursion (3.7) (3.8) we now see that Lemma 3.1 is just an “array part” [or matrix interpretation] of Theorem 4.2. Equivalently, the two schemes (4.19) and (3.5) are the same, so that the fast GEPP algorithm of Sec. 3.5 is the algorithm we are looking for, i.e., a fast algorithm for factorization of rational matrix functions. As we shall see in Sec. 6.5, 6.6, 7.5 pivoting is crucial in achieving high relative accuracy of this factorization, so we next provide an interesting interpolation interpretation for pivoting.

**4.6. Interpolation and fast implementation of pivoting.** Now let us turn to the simplest case of  $W_1(z) = I + C_\pi(I - A_\pi)^{-1} R_1^{-1} B_\zeta$  with simple poles and zeros, i.e., the matrices  $A_\zeta = \text{diag} [x_1 \cdots x_n]$  and  $A_\pi = \text{diag} [y_1 \cdots y_n]$  are diagonal, so that  $R_1$  in  $R_1 A_\pi - A_\zeta R_1 = B_\zeta C_\pi$  is a Cauchy-like matrix. Theorem 4.2

can be used to factorize  $W_1(z) = \Theta_1(z)W_2(z)$ , where the first factor has just one zero  $x_1$  and one pole  $y_1$ :

$$\Theta_1(z) = I + C_{\pi,1} \frac{1}{z - y_1} \frac{1}{r_{11}} B_{\zeta,1}, \quad \text{where} \quad r_{11}y_1 - x_1r_{11} = B_{\zeta,1}C_{\pi,1},$$

i.e., with  $C_{\pi,1}$  be the 1-st column of  $C_\pi$ , and  $B_{\zeta,1}$  be the 1-st row of  $B_\zeta$ . We could, of course just proceed with  $W_2(z)$  recursively to compute in  $O(n^2)$  operations the cascade decomposition (4.11). However, in [GO93], [GO94b] this algorithm was found to be, in general, inaccurate [it even breaks down if  $R_1$  encounters singular leading submatrices].

As one of the methods to overcome these difficulties it was suggested in [GO94b] to apply a factorization step to the same  $W_1(z)$ , but now rewritten in the form

$$W_1(z) = I + C_\pi(I - A_\pi)^{-1}(PR)^{-1}(PB_\zeta),$$

which now produces a different first factor,

$$\Theta_1(z) = I + C_{\pi,1} \frac{1}{z - y_1} \frac{1}{r_{k1}} B_{\zeta,k}, \quad \text{where} \quad r_{k1}y_1 - x_kr_{k1} = B_{\zeta,k}C_{\pi,1},$$

with the same pole  $y_1$  but now with a different zero  $x_k$ . The conclusion is that at each factorization step we have freedom in choosing the zero for the next factor in factorization, but the task is to suggest a heuristic to exploit this freedom to improve accuracy.

Now, the association in theorem 4.2 of factorization of  $W(z)$  and GE procedure was used in [GO94b] to suggest to mimic partial pivoting on  $R_1$  to improve the accuracy of factorization of rational matrix functions, and of many associated algorithms for rational matrix interpolation problems.

Finally note that we essentially used here the diagonal structure only for the “null” matrix  $A_\zeta$ , and not for the “pole” matrix  $A_\pi$ . Hence partial pivoting can be efficiently implemented as well for the case when the “pole” matrix  $A_\pi$  has a more general form, say the Jordan form. This case corresponds to the multiplicity of the poles of  $W_1(z)$  [or, equivalently to the confluence in the null-pole coupling matrix  $R_1$ ], but for brevity we omit these details here.

**4.7. A problem: loss of symmetry for J-unitary rational matrix functions.** In many applications, however [see, e.g., Sec. 6], rational matrix functions and their null-pole coupling matrices enjoy certain symmetry properties. For example we often study *J-unitarity on the imaginary line* functions of the form

$$(4.20) \quad W_1(z) = I - C_\pi(zI - A_\pi)^{-1}R_1^{-1}C_\pi^*J,$$

where  $R_1$  is a Hermitian matrix satisfying

$$(4.21) \quad A_\pi^*R_1 + R_1A_\pi = -C_\pi^*JC_\pi,$$

see Ch. 6 in [BGR90], or [GO94b]. This is a *continuous-time* version, and its *discrete-time* counterpart encounters *J-unitarity on the unit circle* rational functions of the form

$$W_1(z) = [I + C_\pi(zI - A_\pi)^{-1}R_1^{-1}(A_\pi^{-1})^*C_\pi^*J] \cdot D_\beta,$$

where  $R$  is a Hermitian matrix satisfying

$$R_1 - A_\pi^*R_1A_\pi = -C_\pi^*JC_\pi,$$

and

$$D_\beta = I + C_\pi R_1^{-1}(I - \beta A_\pi^*)^{-1}C_\pi^*J,$$

see, e.g., [BGR90], Ch. 7. We shall be interested in accurate and fast algorithms to factorize such  $W_1(z)$ , and of course, general algorithms of Sec. 4.4, 4.5 are immediately applicable. However, they have the following disadvantage. Since symmetry in  $R_1$  is preserved under Schur complementation (2.10), it is easy to see that the symmetry of  $W_1(z)$  shown, e.g., in (4.20) and (4.21) is also preserved under the factorization  $W_1(z) = \Theta_1(z) \cdot W_2(z)$  described in theorem 4.2, see, e.g., (4.17). This means that not only the given function  $W_1(z)$  but also both its factors  $\Theta_1(z), W_2(z)$  are  $J$ -unitary as well. This observation leads to an immediate computational advantage: since  $B_\zeta = JC_\pi^*$  we can run the recursion only on one matrix  $C_\pi$  in (6.6), i.e., to save half of the arithmetic operations and half of memory locations. Unfortunately, without pivoting this scheme suffers of losses of accuracy. Partial pivoting improves accuracy, but unfortunately it destroys symmetry, so it does not lead to savings in speed and memory. This difficulty will be resolved in Sec 6.1 after we shall show next how to implement symmetry-preserving diagonal pivoting techniques.

## 5. Diagonal diagonal pivoting for partially reconstructible Hermitian matrices

As we just noted in Sec. 4.7 a symmetry-preserving accurate algorithms for factorizing a Hermitian structured matrix  $R$  can be obtained by replacing partial pivoting by a fast implementation of diagonal pivoting technique of Sec. 3.5. Although such a replacement seems to cause no technical difficulties, as we shall see in a moment, the symmetric case requires a more scrupulous treatment, because of the following reason.

In the literature on generalized Schur algorithms the displacement operator  $\nabla_A(\cdot)$  is usually assumed to be invertible. However, in applications and for many purposes one may need triangularization algorithms for the more delicate case when displacement operator has a non-trivial kernel [such as, e.g., in [CK91] considering ordinary Hankel matrices]. In this paper we shall meet this necessity in Sec. 7 [see also [KO97b]] when designing more accurate Hermitian Toeplitz and Toeplitz-plus-Hankel solvers. This degeneracy condition for  $\nabla_A(\cdot)$  naturally appears in the context of *boundary* tangential interpolation problems [KO97b], as well as of matrix Nehari and Nehari-Takagi interpolation problems [GO93], [GO94b] and Sec. 6.5, 6.6.

We follow Georg Heinig's suggestion and associate the name "*partially reconstructible*" with the structured matrices associated with degenerate displacement operators.

**5.1. Generators for partially reconstructible matrices.** If the displacement operator

$$(5.1) \quad \nabla_A(R) = A^*R + RA = GJG^*,$$

is invertible, then the  $\nabla_A$ -generator  $\{G, J\}$  on the right-hand-side of (5.1) contains all the information on  $R$ , justifying its name as a generator of  $R$ . In this section we follow [KO97b] to discuss a more peculiar case, when  $\mathcal{K} = \text{Ker } \nabla_A(\cdot)$  is assumed to be nontrivial. In the latter situation a matrix  $R$  with displacement structure (5.1) will be called a *partially reconstructible* matrix, because only part of the information

on  $R$  is now contained in its  $\nabla_A$ -generator  $\{G, J\}$ . To extend the definition of a  $\nabla_A$ -generator to partially reconstructible matrices, let us represent

$$(5.2) \quad R = R_{\mathcal{K}} + R_{\mathcal{K}^\perp}$$

with respect to the orthogonal decomposition  $\mathbf{C}^{n \times n} = \mathcal{K} \oplus \mathcal{K}^\perp$ . A partially reconstructible matrix  $R$  is uniquely determined by the three matrices  $\{G, J, R_{\mathcal{K}}\}$  in (5.1), (5.2), so we shall call them a  $\nabla_A$ -generator of  $R$ . To complete the above definition, one should specify for  $\mathbf{C}^{n \times n}$  the inner product in which the decomposition (5.2) is orthogonal. Here we choose

$$(5.3) \quad \langle A, B \rangle = \text{tr}(B^* \cdot A), \quad A, B \in \mathbf{C}^{n \times n}.$$

where  $\text{tr}(A)$  denotes the sum of all diagonal entries of  $A$ . Note that the latter inner product induces the Frobenius norm in  $\mathbf{C}^{n \times n}$   $\langle A, A \rangle = \sum_{i,j=1}^n |a_{ij}|^2 = \|A\|_F^2$ .

### 5.2. Generators for partially reconstructible Cauchy-like matrices.

When  $\mathcal{K} = \text{Ker } \nabla_A = \{0\}$ , all the information about the  $n^2$  entries of  $R$  is efficiently stored in only  $an + 2$  entries of  $\{G, J\}$ . However if  $\mathcal{K} = \text{Ker } \nabla_A$  is nontrivial, then  $R_{\mathcal{K}}$  is a full  $n \times n$  matrix, and at first glance the representation of a partially reconstructible matrix  $R$  by its generator  $\{G, J, R_{\mathcal{K}}\}$  is no longer efficient. As we shall see in a moment, for the main structured classes,  $\mathcal{K} = \text{Ker } \nabla_A$  has a low dimension, and moreover, the matrix  $R_{\mathcal{K}}$  has a simple form [say, circulant for Toeplitz-like structure, or diagonal matrix for Cauchy-like structure], which in turn can be described by a smaller number of parameters. For example, for Cauchy-like matrices  $R$  we use  $A = \text{diag}(a_1, \dots, a_n)$  so that

$$\mathcal{K} = \text{Ker } \nabla_A = \{\text{diag}(d_1, \dots, d_n)\} \quad \text{where} \quad \begin{cases} d_k \text{ is arbitrary} & \text{if } a_k + a_k^* = 0 \\ d_k = 0 & \text{if } a_k + a_k^* \neq 0 \end{cases} .$$

So the last matrix in the  $\nabla_A$ -generator  $\{G, J, R_{\mathcal{K}}\}$  is a diagonal matrix

$$(5.4) \quad R_{\mathcal{K}} = \text{diag}(d_1, \dots, d_n) \quad \text{where} \quad \begin{cases} d_k = r_{kk} & \text{if } a_k + a_k^* = 0 \\ d_k = 0 & \text{if } a_k + a_k^* \neq 0 \end{cases}$$

i.e., it is formed from the diagonal entries  $\{r_{kk}\}$  of  $R$ .

Similarly, a generator for partially reconstructible Hermitian Toeplitz-like and Toeplitz-plus-Hankel-like matrices will be specified later when they will be needed in Sec. 7.4 and Sec. 7.7.2.

### 5.3. Symmetric GE for partially reconstructible Cauchy-like matrices.

Similarly to the non-Hermitian case (3.4), (3.5), here we need a fast implementation for the block symmetric Schur complementation

$$(5.5) \quad \begin{array}{ccc} R_1 & \rightarrow & R_2 & \rightarrow & \cdots \\ \downarrow & & \downarrow & & \downarrow \\ \{R_{12}^{(1)}(R_{11}^{(1)})^{-1}, R_{11}^{(1)}\} & & \{R_{12}^{(2)}(R_{11}^{(2)})^{-1}, R_{11}^{(2)}\} & & \cdots \end{array}$$

for the case when  $R_1$  is partially reconstructible. Modifying the designation in (2.10) here we use the uper script  $^{(k)}$  to designate the block partitioning for the successive Schur complements  $R_k$ . Instead of the expensive  $O(n^3)$  computation in (5.5), the desired fast algorithm should implement a *block* generator recursion

$$(5.6) \quad \begin{array}{ccccccccc} \{G_1, J, R_{\mathcal{K}_1}\} & \rightarrow & \{G_2, J, R_{\mathcal{K}_2}\} & \rightarrow & \cdots & & & & \\ \downarrow & & \downarrow & & & & & & \\ \{R_{12}^{(1)}(R_{11}^{(1)})^{-1}, R_{11}^{(1)}\} & & \{R_{12}^{(2)}(R_{11}^{(2)})^{-1}, R_{11}^{(2)}\} & & & & & & \cdots \end{array}$$

Here we denote by  $R_{\mathcal{K}_k}$  the the third matrix in a generator  $\{G_k, J, R_{\mathcal{K}_k}\}$  of the *partially reconstructible* matrix  $R_k$ . Recall that  $R_{\mathcal{K}_k}$  is a diagonal matrix in the Cauchy-like case, see (5.4).

In fact, the desired fast algorithm for (5.6) has already been specified in the same Lemma 3.1. Indeed, its generator recursion (3.8) has two properties we need:

- It was formulated for the block Schur complementation step;
- it has been prove without any assumption on the invertibility of the corresponding displacement operator.

[we shall see later in Sec. 6.4 an interesting interpolation interpretation [GO93], [GO94b] for these two conditions].

Thus, Lemma 3.1 is applicable and it suggests a recursion for  $G_k$ . Therefore the only point to clarify here is how to implement the recursion for the diagonal matrices  $R_{\mathcal{K}}^{(k)}$  in  $\{G_k, J, R_{\mathcal{K}}^{(k)}\}$  for  $k = 2, 3, \dots, n$ . Since  $R_{\mathcal{K}_k}$  describes an “unstructured” part of  $R_k$  there seems to be no other choice but to directly apply the structure-ignoring formula (2.10), however using it to compute *only* the first diagonal entries of  $R_k$  which are also the entries  $R_{\mathcal{K}_k}$ , see (5.4) [in the context of interpolation such diagonal entries were called *coupling numbers* by Andre Ran, see [GO94b] and references therein]. Fortunately this structure-ignoring computation will not slow down our algorithm, because at each recursion step we need to compute not more than  $n$  such diagonal entries. This leads us to the following algorithm, in which we do not specify a particular diagonal pivoting technique, because this algorithm can incorporate symmetric or Bunch-Kaufman pivoting [cf. Sec. 2.5], or their combination, see [KO97b]. Other choices are also possible.

Fast symmetric GE with diagonal pivoting  
for partially reconstructible Cauchy-like matrices

**Input:** A  $\nabla_A$ -generator  $\{G_1, J, R_{\mathcal{K}_1}\}$  of a Cauchy-like matrix  $R_1 \in \mathbf{C}^{n \times n}$  partitioned as in (2.10), where  $G_1 \in \mathbf{C}^{n \times \alpha}$ , and  $R_{\mathcal{K}_1} = \text{diag}(d_1^{(1)}, \dots, d_s^{(1)}, 0, \dots, 0)$ .

**Output:** The block  $LDL^*$  decomposition of  $R_1$ .

**Steps:** 1: Using a particular pivoting technique, perform symmetric row and column interchanges, and determine the size  $m$  for the pivot  $R_{11} \in \mathbf{C}^{m \times m}$ .

For this, recover the necessary entries of  $R_1$  from  $\{G_1, J, R_{\mathcal{K}_1}\}$   $r_{ij} = \frac{g_i J g_j^*}{a_i^* + a_j}$  where  $g_i$  denotes the  $i$ -th row of  $G_1$ .

The algorithm remains fast as long as we need to recover  $O(n)$  entries, e.g., in symmetric and Bunch-Kaufman pivoting.

2: Recover from  $\{G_1, J, R_{\mathcal{K}_1}\}$  the entries of the first  $m$  columns  $\begin{bmatrix} R_{11} \\ R_{21} \end{bmatrix}$  of  $R_1$ .

[Note that some of the entries  $r_{ij}$  can be already available after the step 1.]

3: Store the first  $m \times m$  block diagonal entry  $R_{11}$  of the matrix  $D$  and the first  $m$  columns  $\begin{bmatrix} I \\ R_{21} \cdot R_{11}^{-1} \end{bmatrix}$  of the matrix  $L$  in the  $LDL^*$  decomposition of  $R_1$ .

4: Compute a generator  $\{G_2, J, R_{\mathcal{K}_2}\}$  of the Schur complement  $R_2$ :

4.1: Compute  $G_2 \in \mathbf{C}^{(n-m) \times \alpha}$  via

$$\begin{bmatrix} 0 \\ G_2 \end{bmatrix} = G_1 - \begin{bmatrix} I_m \\ R_{21} R_{11}^{-1} \end{bmatrix} G_{11}.$$

4.2: Compute the diagonal matrix  $R_{\mathcal{K}_2}$  by applying the standard Schur complementation formula  $R_2 = R_{22} - R_{21} R_{11}^{-1} R_{21}^*$  to compute

only nonzero diagonal entries of  $R_2$  [these entries  $d_k^{(2)}$  are those for which we have  $a_k + a_k^* = 0$ , e.g., (5.4). Namely

- if  $m = 1$  then  $d_k^{(2)} = d_k^{(1)} - r_{k1} \frac{1}{r_{11}} r_{k1}^*$ .

- If  $m = 2$  then  $d_k^{(2)} = d_k^{(1)} - [ \begin{array}{cc} r_{k1} & r_{k2} \end{array} ] \cdot \left[ \begin{array}{cc} r_{11} & r_{21}^* \\ r_{21} & r_{22} \end{array} \right]^{-1} \cdot \left[ \begin{array}{c} r_{k1}^* \\ r_{k2}^* \end{array} \right]$ .

**5:** Repeat recursively for the Schur complement  $R_2$ , given by its generator  $\{G_2, J, R_{K_2}\}$ .

**5.4. Transformations between continuous-time and discrete-time displacement operators.** Each pattern of structure in Table 4 can be equivalently defined via different choices of the displacement operator. For example, along with the *continuous-time* displacement operator

$$(5.7) \quad \nabla_A(R) = A^* \cdot R + R \cdot A = G_{\nabla_A} \cdot J \cdot G_{\nabla_A}^*,$$

mainly discussed above, its *discrete-time* counterpart

$$(5.8) \quad \nabla_F(R) = R - F \cdot R \cdot F^* = G_{\nabla_F} \cdot J \cdot G_{\nabla_F}^*,$$

also often appears in applications. Therefore we need versions of fast and accurate algorithms for this important case as well. The next two lemmas indicate how to transform the equations (5.7) and (5.8) to each other without changing the solution  $R$ . These formulas will show how to easily translate a certain algorithm from one form to another. We shall also need this technique later in Sec. 7.5.

**LEMMA 5.1.** *Let  $F \in \mathbf{C}^{n \times n}$  be arbitrary, and set*

$$(5.9) \quad A = (\beta I + F) \cdot (\beta I - F)^{-1},$$

*where  $\beta$  is any number such that  $|\beta| = 1$  chosen to guarantee the invertibility of the second factor in (5.9). Let the displacement operators  $\nabla_F$  and  $\nabla_A$  be defined by (5.8) and (5.7), respectively. Then*

(i):  $\text{Ker } \nabla_F = \text{Ker } \nabla_A$ .

(ii): *For any  $R \in \mathbf{C}^{n \times n}$ , specified by the  $\nabla_F$ -generator  $\{G_{\nabla_F}, J, R_K\}$ , its  $\nabla_A$ -generator is  $\{G_{\nabla_A}, J, R_K\}$ , where*

$$(5.10) \quad G_{\nabla_A} = \sqrt{2} \cdot (\beta I - F)^{-1} G_{\nabla_F}.$$

*In particular, the  $\nabla_F$ -displacement rank and the  $\nabla_A$ -displacement rank of  $R$  are the same.*

**Proof.** Both assertions follow from the identity

$$\begin{aligned} 2 \cdot (\beta I - F)^{-1} \cdot (R - F \cdot R \cdot F^*) \cdot (\beta^* I - F^*)^{-1} = \\ = (\beta I - F)^{-1} \cdot (\beta I + F) \cdot R + R \cdot (\beta^* I + F^*) \cdot (\beta^* I - F^*)^{-1}. \end{aligned}$$

◊

The next lemma is the converse to Lemma 5.1, and it is deduced by exactly the same arguments.

**LEMMA 5.2.** *Let  $A \in \mathbf{C}^{n \times n}$  be arbitrary, and set*

$$(5.11) \quad F = (tI + A)^{-1} \cdot (tI - A),$$

*where  $t > 0$  is any number that will guarantee the invertibility of the second factor in (5.11). Let the displacement operators  $\nabla_F$  and  $\nabla_A$  be defined by (5.8) and (5.7), resp. Then*

(i):  $\text{Ker } \nabla_F = \text{Ker } \nabla_A$ .

(ii): Any  $R \in \mathbf{C}^{n \times n}$  given by the  $\nabla_A$ -generator  $\{G_{\nabla_A}, J, R_K\}$  has a  $\nabla_F$ -generator  $\{G_{\nabla_F}, J, R_K\}$ , where

$$(5.12) \quad G_{\nabla_F} = \sqrt{2t} \cdot (tI + A)^{-1} \cdot G_{\nabla_A}$$

In particular, the  $\nabla_F$ -displacement rank and the  $\nabla_A$ -displacement rank of  $R$  are the same.

**Proof.** Both assertions follow from the easily checked identity

$$\begin{aligned} & 2 \cdot t \cdot (tI + A)^{-1} \cdot (A \cdot R + R \cdot A^*) \cdot (tI + A^*)^{-1} = \\ & = R - (tI + A)^{-1} \cdot (tI - A) \cdot R \cdot (tI - A^*) \cdot (tI + A^*)^{-1}. \end{aligned}$$

◊

**5.5. Algorithm for the discrete-time case.** The above two lemmas show how to translate certain formulas from continuous-time to discrete-time, obtaining the following counterpart of lemma 3.1.

LEMMA 5.3. ([AD93], [KS95b], [KO97b] /see also Sec. 8 in [GO94b]/) Let  $F \in \mathbf{C}^{n \times n}$  be a block lower triangular matrix,  $R_1 \in \mathbf{C}^{n \times n}$  be a Hermitian matrix, satisfying a discrete-time Lyapunov displacement equation of the form

$$(5.13) \quad \nabla_F(R) = R_1 - F \cdot R_1 \cdot F^* = G_1 \cdot J \cdot G_1^*,$$

for some  $G_1 \in \mathbf{C}^{n \times \alpha}$ , and some Hermitian matrix  $J \in \mathbf{C}^{\alpha \times \alpha}$ . Let the matrices in (5.13) be partitioned as

$$F = \begin{bmatrix} F_{11} & 0 \\ F_{21} & F_{22} \end{bmatrix}, \quad R_1 = \begin{bmatrix} R_{11} & R_{21}^* \\ R_{21} & R_{22} \end{bmatrix}, \quad G_1 = \begin{bmatrix} G_{11} \\ G_{21} \end{bmatrix}.$$

If the upper left block  $R_{11} \in \mathbf{C}^{m \times m}$  of  $R_1$  is invertible, then the Schur complement  $R_2 = R_{22} - R_{21} \cdot R_{11}^{-1} \cdot R_{21}^* \in \mathbf{C}^{(n-m) \times (n-m)}$  satisfies

$$(5.14) \quad R_2 - F_{22} \cdot R_2 \cdot F_{22}^* = G_2 \cdot J \cdot G_2^*,$$

with

$$(5.15) \quad G_2 = G_{21} - ((\beta I - F_{22}) \cdot R_{21} \cdot R_{11}^{-1} - F_{21}) \cdot (\beta I - F_{11})^{-1} \cdot G_{11},$$

where  $\beta$  is any number on the unit circle, which is not an eigenvalue of  $F_1$ .

This lemma is easily deduced by applying Lemma 5.2 to change the form of displacement equation to the continuous-time one, then use the Lemma 3.1 to obtain the formula for the generator recursion, and finally to again change the form of displacement equation to the discrete-time one. See, e.g., [KO97b] for details.

We now have several fast displacement-based implementations of the GEPP and of symmetric GE with diagonal pivoting for Cauchy-like matrices [for usual and for partially reconstructible]. Thanks to pivoting these algorithms have a good record of numerically accurate performance in certain applications, see [GO93], [GO94b], and therefore it is attractive to use these algorithms for solving linear equations with coefficient matrices having other [i.e., not only Vandermonde-like or Cauchy-like] patterns of structure listed in Table 4. In Sec. 7 it will be achieved by using the transformation approach. However, before doing so we return to the problem noted in Sec. 4.7 and to clarify how fast algorithms of Sec. 5 with diagonal pivoting factorize  $J$ -unitary rational matrix functions.

## 6. Diagonal pivoting and factorization of $J$ -unitary rational matrix functions

To begin with, let us indicate a class of applications giving rise to  $J$ -unitary functions.

**6.1. Tangential interpolation with norm constraints.** Often one is looking for a rational rectangular  $M \times N$  matrix interpolant  $F(z)$ , satisfying non-homogeneous [as opposed to the example in Sec. 4.2] tangential interpolation conditions

$$(6.1) \quad u_k \cdot F(z_k) = v_k,$$

where  $\{u_k, v_k\}$  are given vectors of appropriate sizes. It is trivial to reduce this problem to the homogeneous one. Indeed, if we have

$$(6.2) \quad \begin{bmatrix} u_k & -v_k \end{bmatrix} \cdot \begin{bmatrix} W_{11}(z_k) & W_{12}(z_k) \\ W_{21}(z_k) & W_{22}(z_k) \end{bmatrix} = \begin{bmatrix} 0 & 0 \end{bmatrix}.$$

then, of course, one  $F(z)$  for (6.1) would be given by

$$(6.3) \quad F(z) = W_{12}(z)W_{22}(z)^{-1}.$$

For physical reasons [i.e., conservation of energy] we are often looking for a *passive* interpolant, imposing the norm constraint

$$(6.4) \quad \|F(z)\| \leq 1$$

for a certain set of point, say on the imaginary line  $i\mathbf{R}$  or on the unit circle. For example, if all the nodes  $\{x_k\}$  are in the interior of the unit disk, and we are looking for a scalar rational passive interpolant  $f(x_k) = v_k$ , then this is the celebrated Nevanlinna-Pick problem [see **[A65]** and references therein], and the Hermitian null-pole coupling matrix is just a Pick matrix  $R = \begin{bmatrix} \frac{1-v_j x_j^*}{1-x_j^* x_j} \end{bmatrix}$ .

Fortunately, such non-homogeneous problems can be reduced to homogeneous ones, so methods of Sec. 4.2, 4.3 and 4.4 are applicable. Indeed, the condition (6.4) for the non-homogeneous interpolant  $F(z)$  can be guaranteed by requiring a certain symmetry for the homogeneous interpolant  $W(z) = \begin{bmatrix} W_{11}(z_k) & W_{12}(z_k) \\ W_{21}(z_k) & W_{22}(z_k) \end{bmatrix}$ . Specifically, we are done if  $W(z)$  is a  $J$ -unitary rational matrix function [formally defined below], whose null-pole coupling matrix  $R$  is Hermitian [ $R$  is even positive definite if we require passivity not only on the boundary, but for the whole region, say, in the half-plane, or in the unit disk].

**6.2.  $J$ -unitary rational matrix functions.** Let  $J$  be a  $\alpha \times \alpha$  signature matrix. A rational  $\alpha \times \alpha$  matrix function  $W(z)$  is called  *$J$ -unitary on the imaginary line  $i\mathbf{R}$*  if it satisfies

$$(6.5) \quad (W(z))^* \cdot J \cdot W(z) = J \quad \text{on} \quad z \in i\mathbf{R}.$$

The point is that in this case the symmetry in  $W(z)$  allows us to represent it using only half of the data [as compared to the general case]. More precisely, if we know only two “pole” matrices  $\{C_\pi, A_\pi\}$  in (4.6), with  $B_\pi$  unknown, then we are done. Specifically [see Ch. 6 in **[BGR90]**, or **[GO94b]**], the global left null-pole triple defined in (4.9) has a lot of redundant information:

$$(6.6) \quad \tau = \{C_\pi, A_\pi, -A_\pi^*, -C_\pi^* J, R\}.$$

so that

$$(6.7) \quad W(z) = I - C_\pi(zI - A_\pi)^{-1}R^{-1}C_\pi^*J,$$

where  $R$  is a Hermitian matrix satisfying

$$(6.8) \quad A_\pi^*R + RA_\pi = -C_\pi^*JC_\pi.$$

**6.3. Diagonal pivoting and factorization of  $J$ -unitary rational matrix functions.** Now one can easily apply the arguments of Sec. 4.5 to see that the fast symmetric GE with diagonal pivoting algorithm of Sec. 5.3 for triangularization of  $R$  in (6.8) also computes the cascade decomposition (4.11) for the corresponding  $J$ -unitary on the imaginary line rational matrix function  $W(z)$  in (6.7). Moreover, all factors  $\Theta_k(z)$  in this decomposition are  $J$ -unitary as well, and this algorithm is twice as fast as compared to the fast GEPP of Sec. 3.5, 4.4, 4.5.

We conclude this section by giving an interesting interpolation interpretation to the two comments made just before formulating the algorithm in Sec. 5.3.

- **Block elimination steps.** It is well-known that it is not always possible to decompose a  $J$ -unitary rational matrix function into a product of the first-order  $J$ -unitary factors [i.e., ones having just one pole and zero]. It was shown in [AD86] that fortunately, it is always possible to write down a cascade decomposition for a  $J$ -unitary rational function with at most second-order factors [i.e., having two poles and two zeros]. In fact, in the Bunch-Kaufman pivoting technique we apply either scalar or block  $2 \times 2$  block elimination steps, see Sec. 2.5. A single elimination step on  $R_1$  corresponds to writing down a first-order factor in the cascade decomposition (4.11), whereas a  $2 \times 2$  block elimination step corresponds to writing down a second-order factor. Clearly, if the (1,1) entry of  $R_1$  is zero, than we cannot perform a GE step, and if the whole diagonal of  $R_1$  zero, than the situation cannot be fixed with symmetric row/column interchanges, and we have to perform a block elimination step. In fact, the Bunch-Kaufman pivoting technique performs such  $2 \times 2$  block elimination steps not only in the case of exactly zero diagonal, but for sufficiently small pivots as well.
- **Displacement operator with nontrivial kernels.** A rational scalar function cannot have a pole and a zero at the same point; this is obviously possible with rational *matrix* functions. A closer look at (4.7) [where the eigenvalues of  $A_\pi$  are the poles, and the eigenvalues of  $A_\zeta$  are the zeros of  $W(z)$ ] shows that this situation gives rise to a displacement operator with a non-trivial kernel, and to what we have called partially reconstructible matrices in Sec. 5. In fact, rational matrix functions with the zeros and poles at the same points appear in many problems. For example, in *boundary tangential interpolation problems* we may be given the interpolation data  $\{x_k, u_k, v_k\}$  [i.e., corresponding to an unknown function  $F(z)$  in (6.1)], but now with the points not only inside the right-half plane, but also on its boundary  $i\mathbf{R}$ . In this case we shall also reduce the problem to the homogeneous one as in (6.2), and again the nodes  $x_k$  will be the zeros of  $W(z)$ . The point is that since  $W(z)$  will be  $J$ -unitary, its zeros on  $i\mathbf{R}$  should also be its poles, see, e.g., (6.5). To sum up, the

boundary nodes  $x_k \in i\mathbf{R}$  give rise to a displacement operator with non-trivial kernel, as can be seen directly from (6.8). We refer to [KO97b] for the details.

Another example of such a problem is the rational matrix Nehari interpolation problem, discussed in Sec. 6.5, 6.6 below.

**6.4. Discrete-time case.** We have discussed the case of a  $J$ -unitary on the line rational matrix function, which is the *continuous-time* version of the problem. The *discrete-time* version corresponds to the  *$J$ -unitarity on the unit circle*  $\partial\mathcal{D}$  rational  $\alpha \times \alpha$  matrix functions  $W(z)$ , i.e., those satisfying  $W(z) \cdot J \cdot (W(\frac{1}{z^*}))^* = J$  on  $|z| = 1$ . In the continuous-time case above we assumed  $W(z)$  to have the identity value at infinity. Here, in the discrete-time case, we shall assume instead that  $W(\beta) = I$ , where  $\beta$  is a certain point on the unit circle,  $|\beta| = 1$ . Again, the left global left null-pole triple has a lot of redundant information. As above, if we are given only two “pole matrices,”  $\{C_\pi, A_\pi\}$ , they already define the whole triple  $\tau = \{C_\pi, A_\pi, (A_\pi^{-1})^*, (A_\pi^{-1})^* C_\pi^* J, R\}$ , so that

$$W(z) = [I + C_\pi(zI - A_\pi)^{-1} R^{-1} (A_\pi^{-1})^* C_\pi^* J] \cdot D_\beta,$$

where  $D_\beta = I + C_\pi R^{-1} (I - \beta A_\pi^*)^{-1} C_\pi^* J$ , and  $R$  is a Hermitian matrix satisfying

$$(6.9) \quad R - A_\pi^* R A_\pi = -C_\pi^* J C_\pi.$$

see, e.g., [BGR90], Ch. 7.

Now recall that Lemma 5.3 describes the displacement structure of the Schur complement for the case when this displacement structure is defined via the discrete-time displacement operator, such as in (6.9). Therefore this Lemma 5.3 is just an “array part” of the corresponding version of theorem on factorization of rational  $J$ -unitary on the unit circle matrix functions.

We conclude this subsection with noting that these two versions of a factorization theorem can be easily deduced from each other, by providing an interpolation interpretation for the proof of Lemma 5.3. Recall that it was based on the results of Sec. 5.4, and consisted of three steps. First transform the discrete-time displacement equation (6.9) to a continuous-time one (6.8), then perform one step of Schur complementation using Lemma 3.1, and finally transform the obtained formulas back to the discrete-time form. In the function domain these three steps are even simpler: we are given a  $J$ -unitary on the circle rational matrix function  $W_1(z)$ , which we would like to factorize. In the first step we choose an appropriate Möbius transformation  $\varphi(z) = \beta \frac{z-1}{z+1}$  of the unit circle onto the right-half-plane, so that  $W_1(\beta \frac{z-1}{z+1})$  is now a  $J$ -unitary on the  $i\mathbf{R}$  rational matrix function. Therefore, in the second step we can use theorem 4.2 to factorize it,

$$W_1(\beta \frac{z-1}{z+1}) = \Theta_1(\beta \frac{z-1}{z+1}) \cdot W_2(\beta \frac{z-1}{z+1}).$$

Finally, in the third step we map the right-half plane back to the unit circle, obtaining formulas in Lemma 5.3, see, e.g., [KO97b] for details. This reinforces the point made in Sec. 5.4 that there are no essential differences between different forms for the displacement operator, but it may be important to provide a user with a particular form of a theorem or of an algorithm he may need.

**6.5. Numerical example: Nehari problem.** There are many interpolation problems, e.g., listed in Sec. 4.1, for which we are able to specify the above results of Sec. 6. We present only one such example, for which it was shown numerically in [GO94b] that partial and diagonal pivoting techniques indeed lead to the higher accuracy. Loosely speaking, in this problem we are given a rational matrix function  $K(z)$

- with several poles in the left-half plane,
- bounded on the imaginary line:  $\sup_{z \in i\mathbf{R}} \|K(z)\| < \infty$ ,

and our task is to fix it, i.e., to find a new function  $K(z) - R(z)$

- with the same pole structure in the left-half-plane [i.e., the fixing term  $R(z)$  is analytic there],
- passive on the imaginary line:  $\sup_{z \in i\mathbf{R}} \|K(z) - R(z)\| < 1$ .

Here is the precise formulation of this problem.

Suboptimal rational matrix Nehari problem. Simple poles.

Given:  $n$  complex points  $\{z_1, \dots, z_n\}$  in the left-half-plane  $\Pi^-$ ,  
 $n$   $1 \times N$  row vectors  $\{w_1, \dots, w_n\}$ ,  
 $n$   $M \times 1$  column vectors  $\{\gamma_1, \dots, \gamma_n\}$ ,

and a rational  $M \times N$  matrix function  $K(z)$  having only  $n$  simple poles  $\{z_k\}$  in the left-half-plane, with a local Laurent series in a neighborhood of each  $z_k$ :

$$K(z) = (z - z_k)^{-1} \gamma_k \cdot w_k + \text{[analytic at } z_k \text{]} .$$

such that

$$\sup_{z \in i\mathbf{R}} \|K(z)\| < \infty.$$

Find: A rational  $M \times N$  matrix function  $R(z)$  with no poles in  $\Pi^- \cup i\mathbf{R}$ , such that

$$\sup_{z \in i\mathbf{R}} \|K(z) - R(z)\| < 1.$$

Solvability condition: The  $2n \times 2n$  matrix

$$(6.10) \quad R = \begin{bmatrix} Q & I \\ I & P \end{bmatrix}$$

has exactly  $n$  positive and  $n$  negative eigenvalues, [equivalently,  $\lambda_1(PQ) < 1$ ], where

$$P = \left[ -\frac{w_i \cdot w_j^*}{z_i + z_j^*} \right], \quad Q = \left[ -\frac{\gamma_i^* \cdot \gamma_j}{z_i^* + z_j} \right]$$

Solution: , as given, e.g., in [BGR90]:

$$K(z) - R(z) = [W_{11}(z)G(Z) + W_{12}(z)] \cdot [W_{21}(z)G(z) + W_{22}(z)]^{-1},$$

with an arbitrary parameter  $G(z)$  satisfying

- $G(z)$  has no poles in  $\Pi^- \cup i\mathbf{R}$ ,
- $\sup_{z \in i\mathbf{R}} \|G(z)\| \leq 1$ ,

and

$$(6.11) \quad \boxed{W(z) = \begin{bmatrix} W_{11}(z) & W_{12}(z) \\ W_{21}(z) & W_{22}(z) \end{bmatrix} = I_{M+N} - C_\pi(zI_{2n} - A_\pi)^{-1} R^{-1} C_\pi^* J},$$

where  $R$  is given by (6.10),  $J = \begin{bmatrix} I_N & 0 \\ 0 & -I_M \end{bmatrix}$ , and

$$A_\pi = \text{diag}(z_1, \dots, z_n, -z_1^*, \dots, -z_n^*), \quad C_\pi = \begin{bmatrix} \gamma_1 & \dots & \gamma_n & 0 & \dots & 0 \\ 0 & \dots & 0 & w_1^* & \dots & w_n^* \end{bmatrix}.$$

The solution (6.11) for the Nehari problem is yet another example of using the same BGR formula (4.8) which is not convenient because it involves the inverse of  $R$ . Therefore we might reinforce the comment made in Sec. 4.2. Specifically, we have a complete description for the solution of the Nehari problem, but we still have to design efficient accurate algorithm to compute this solution. Of course,  $R$  in (6.10) is a partially reconstructible Cauchy-like matrix satisfying

$$(6.12) \quad A_\pi^* R + R A_\pi = -C_\pi^* J C_\pi,$$

The given data does not specify the entries of  $R$ , and it is given by its generator  $\{C_\pi, -J, R_K\}$  [see Sec. 5.1 for the definition of a generator of partially reconstructible matrices]. The first two matrices of this generator are provided by the right-hand-side of (6.12), and the third matrix has the form  $R_K = \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix}$ , because the kernel of  $\nabla_A$  is clearly the set of matrices of the form  $\begin{bmatrix} 0 & D_1 \\ D_2 & 0 \end{bmatrix}$  with arbitrary diagonal  $D_1, D_2$ .

Given this generator  $\{C_\pi, -J, R_K\}$ , we are able to run the fast symmetric GE with diagonal pivoting algorithm of Sec. 5.3 to rapidly compute the block  $LDL^*$  factorization of  $R$  in  $O(n^2)$  operations. The rational matrix function  $W(z)$  given by (6.11) is  $J$ -unitary on the imaginary line. As was explained in Sec. 6.3, the same algorithm of Sec. 5.3 rapidly computes the cascade decomposition (4.11) for  $W(z)$ . Having this cascade decomposition we are able to evaluate  $W(z)$  [and therefore  $K(z) - R(z)$  as well] in linear time.

In a wide variety of computed examples in [GO94b] it was found that indefinite matrix  $R$  [recall that it has exactly  $n$  positive and  $n$  negative eigenvalues] is usually moderately conditioned. However, its blocks  $P$  and  $Q$  are positive definite matrices that were always extremely ill-conditioned. Therefore it is not surprising that numerical examples led us to the conclusion that partial and diagonal pivoting techniques are needed to compute not only fast, but also accurate, factorization. We refer to [GO94b] for many numerical experiments and for practical recommendations. Here we only include one illustration with a  $2 \times 2$  rational matrix function given by the interpolation data at 60 interpolation points [so  $R$  is  $120 \times 120$  matrix]. The solvability condition [which is  $\lambda_1(PQ) < 1$ ] is almost violated:  $\lambda_1(PQ) = 0.99$ . Table 6.5 shows the relative forward error  $\frac{\|F(z_0) - \hat{F}(z_0)\|_\infty}{\|F(z_0)\|_\infty}$  for the evaluation of the solution  $F(z) = K(z) - R(z)$  at a certain point  $z_0$ . As usual, the hatted quantity denotes the actually *computed* result.

In this case all the fast methods produce the same accuracy as the much slower GECP [we found that partial pivoting is often slightly better than diagonal pivoting]. There are classes of problems for which fast algorithms are even more accurate than the slow GECP, as illustrated in the next section.

**6.6. Numerical Example: Nehari-Takagi problem.** We again recall the solvability condition for the matrix Nehari problem:  $\lambda_1(PQ) < 1$ , or equivalently,

Fast Cascade decomposition			Fast Solving (4.10)			Slow Structure- ignoring GECP
No pivoting	Partial pivoting	Diagonal pivoting	No pivoting	Partial pivoting	Diagonal pivoting	
4.4e+00	3.2e-05	9.2e-05	2.4e+00	1.3e-05	6.1e-05	2.4e-05

TABLE 5. *Forward relative error for a matrix Nehari problem.*

Fast Cascade decomposition			Fast Solving (4.10)			Slow Structure- ignoring GECP
No pivoting	Partial pivoting	Diagonal pivoting	No pivoting	Partial pivoting	Diagonal pivoting	
1.5e+00	2.2e-02	2.5e-05	6.7e+00	2.1e-02	2.6e-03	7.5e-02

TABLE 6. *Forward relative error for a matrix Nehari-Takagi problem.*

the null-pole coupling matrix  $R = \begin{bmatrix} Q & I \\ I & P \end{bmatrix}$  has the inertia  $(n, n, 0)$  [i.e., exactly  $n$  positive and exactly  $n$  negative eigenvalues]. The more general Nehari-Takagi problem deals with the situation where this solvability condition is violated, and the inertia of  $R$  is, say  $(n+s, n-s, 0)$ . In this case we cannot fix our given  $K(z)$ , i.e. to achieve the desired inequality

$$\|K(z) - R(z)\| \leq 1 \quad z \in i\mathbf{R},$$

with the fixing function  $R(z)$  analytic in  $\Pi^-$  [as it was in the Nehari problem], but the point is that we can now fix  $K(z)$  with  $R(z)$  having only  $s$  poles in  $\Pi^-$ . Moreover, all the formulas in Sec. 6.5, and therefore all the algorithms remain valid. Interestingly, such slightly different eigenvalue profile causes different numerical properties for the algorithms discussed. Specifically, it was found that diagonal pivoting now performs better than partial pivoting, and that fast algorithms provide higher accuracy than the method based on the structure-ignoring GECP. Again we illustrate these conclusions only with one example, and refer to [GO94b] for more information. The following table shows the relative forward error for a  $2 \times 2$  matrix function, given by the interpolation data at 60 interpolation points [so  $R$  is  $120 \times 120$  matrix], and we allow  $R(z)$  to have 15 simple poles.

Given the evidence that fast algorithms with pivoting for Cauchy-like matrices are accurate, it is attractive to extend this approach to solve linear systems with other patterns of structure. One way of doing this is described next.

## 7. Toeplitz and Toeplitz-plus-Hankel solvers: transformations and pivoting

**7.1. Transformation-and-pivoting.** As we noted in (3.10), fast implementation of partial pivoting are possible for all those structures in Table 4 that are defined using diagonal matrices  $F$  for the corresponding displacement operators

$$(7.1) \quad \nabla_{\{F,A\}}(R) = FR - RA = GB$$

[cf. with the comment made after (3.10)]. The question is: how to devise accurate fast algorithms for the other important kinds of displacement structure in Table 4,

including Toeplitz-like, Hankel-like, Toeplitz-plus-Hankel-like, Chebyshev-Hankel-like, polynomial-Hankel-like matrices for which  $F$  not diagonal. One way of doing this is to exploit a useful observation that these matrices  $F$  can be diagonalized.  $F = T_F^{-1}D_FT_F$ ,  $A = T_A^{-1}D_AT_A$ , so that the matrix  $T_FRT_A^{-1}$  is a Cauchy-like matrix satisfying

$$D_F(T_FRT_A^{-1}) - (T_FRT_A^{-1})D_A = (T_FG)(BT_A^{-1}).$$

So its permuted triangular factorization,  $T_FRT_A^{-1} = PLU$ , can be computed via the fast GEPP algorithm of Sec. 3.5. Of course, all the computation can be done as a manipulation on generators, as seen from the following scheme of the overall algorithm for solving  $Ra = f$ .

A scheme for transformation-and-pivoting algorithms for solving  $Ra = f$

**INPUT:** A  $\nabla_{\{F,G\}}$ -generator  $\{G, B\}$  in (7.1) and  $f$ .

**OUTPUT:** The solution of  $Ra = f$ .

- (1) Compute the  $\nabla_{\{D_F, D_A\}}$ -generator

$$(7.2) \quad \{T_FG, BT_A^{-1}\}$$

for the Cauchy-like matrix  $T_FRT_A^{-1}$ .

- (2) Use this generator as the input for the fast GEPP algorithm in Sec. 3.5 to compute the permuted triangular factorization  $T_FRT_A^{-1} = PLU$ .
- (3) Solve

$$(7.3) \quad (PLU)b = T_Ff$$

via the forward and back-substitution.

- (4) Finally, compute the solution.

$$(7.4) \quad a = T_A^{-1}b.$$

As we shall see in a moment, in all cases interesting to us, matrices  $T_F$ , and  $T_A$  will be certain discrete transform matrices, for which accurate superfast  $O(n \log n)$  algorithms are available for multiplication and solving linear system. Therefore the algorithm just specified will have the same complexity  $O(n^2)$  operations as the fast GEPP algorithm of Sec. 3.5 used in the step 2 of the above scheme. Indeed, the only difference will be  $O(n \log n)$  operations to compute  $2(\alpha+1)$  discrete transforms needed in (7.2), (7.3) and (7.4). Here  $\alpha$  is the displacement rank, i.e., the number of columns of each of matrices  $\{G, B^T\}$ .

We next specify this scheme for Toeplitz-like and Toeplitz-plus-Hankel-like matrices.

**7.2. Toeplitz-like matrices. FFT-based transformation.** *Toeplitz-like* matrices have low displacement rank with respect to the displacement operator (7.5)

$$\nabla_{\{Z_1, Z_{-1}\}}(R) = Z_1R - RZ_{-1} = GB, \quad \text{where} \quad Z_\phi = \begin{bmatrix} 0 & 0 & \cdots & 0 & \phi \\ 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix}.$$

Now, for arbitrary Toeplitz-like matrix the general scheme of Sec. 7.1 can be nicely implemented, because  $Z_\phi$  in (7.5) is a  $\phi$ -circulant matrix, so it is diagonalized by the scaled discrete Fourier matrix  $\mathbf{F} = \frac{1}{\sqrt{n}}(\omega^{ij})_{i,j=0}^{n-1}$  [with  $\omega$  denoting the primitive  $n$ -th root of unity]:

$$(7.6) \quad Z_\phi = D_\phi^{-1} \cdot \mathcal{F}^* \cdot D_{Z_\phi} \cdot \mathcal{F} \cdot D_\phi,$$

with  $D_{Z_\phi} = \text{diag}((\xi\omega^i)_{i=0}^{n-1})$ , and  $D_\phi = \text{diag}((\xi^i)_{i=0}^{n-1})$ , where  $\xi$  is an arbitrary complex number satisfying  $\xi^n = \phi$ . Summarizing, for Toeplitz-like matrices the transformation-and-pivoting algorithm of Sec. 7.1 is based on Fast Fourier transform (FFT), because in this case we have:

$$T_{Z_1} = \mathcal{F}, \quad T_{Z_{-1}} = \mathcal{F}D_{-1}.$$

Note that the both above similarity matrices are unitary, and that in this case the transformed Cauchy-like matrix  $\mathcal{F}RD_{-1}^*\mathcal{F}^*$  in

$$D_1(\mathcal{F}RD_{-1}^*\mathcal{F}^*) - (\mathcal{F}RD_{-1}^*\mathcal{F}^*)D_{-1} = (\mathcal{F}G)(BD_{-1}^*\mathcal{F}^*)$$

is very special, with the nodes [i.e., diagonal entries of  $D_1, D_{-1}$ ] being primitive  $n$ -th roots of 1 and  $-1$ , resp.

We refer to [GKO95] for a large number of computed examples showing that this transformation-and-pivoting method [called the GKO algorithm] in practice provides a high relative forward and backward accuracy.

**7.3. A problem: loss of symmetry. Partially reconstructible matrices.** The weakness of the approach in Sec. 7.2 is that it ignores the possible symmetry in  $R$ . Indeed, we used (7.5) to define the class of Toeplitz-like matrices, so even for Hermitian  $R$  we have two different generator matrices  $\{G, B\}$ . As a result, a transformed Cauchy-like matrix,  $\mathcal{F}RD_{-1}^*\mathcal{F}^*$ , is no longer Hermitian.

To describe a symmetry-preserving transformation-and-pivoting algorithm we exploit the fact that there are many equivalent forms of displacement equations for Toeplitz-like matrices. For example, for Hermitian Toeplitz-like matrices  $R$  we can equivalently use the  $\phi$ -cyclic displacement equation

$$(7.7) \quad \nabla_{Z_\phi}(R) = R - Z_\phi \cdot R \cdot Z_\phi^* = G \cdot J \cdot G^*,$$

where  $J$  is a  $\alpha \times \alpha$  signature matrix, so the structure of  $R$  is now captured by the  $\alpha n$  entries of only one generator matrix  $G$ . For any  $|\phi| \neq 1$  the identity (7.6) can be used to transform a *Hermitian* Toeplitz-like matrix  $R$  into a *Hermitian* Cauchy-like matrix  $\mathcal{F}D_\phi RD_\phi^*\mathcal{F}^*$ , satisfying

$$(7.8)$$

$$\nabla_{\Lambda_\phi}(\mathcal{F}D_\phi RD_\phi^*\mathcal{F}^*) = (\mathcal{F}D_\phi RD_\phi^*\mathcal{F}^*) - \Lambda_\phi \cdot (\mathcal{F}D_\phi RD_\phi^*\mathcal{F}^*) \cdot \Lambda_\phi^* = (\mathcal{F}D_\phi G) \cdot J \cdot (\mathcal{F}D_\phi G)^*.$$

Then instead of fast GEPP of Sec. 3.5 [which would destroy the Hermitian character of the Cauchy-like matrix  $\mathcal{F}D_\phi RD_\phi^*\mathcal{F}^*$ ] we can use the fast symmetric GE with diagonal pivoting of Sec. 5.5 or of Sec. 5.3 [see [KO97b] for details]. For *any* choice of  $\phi$  in (7.7) the resulting algorithm is about twice as fast as the original GKO algorithm.

The problem is, that while the different choices of  $\phi$  are all theoretically equivalent, the corresponding computational schemes have different numerical properties. Our numerical experiments indicate that choices with  $|\phi| \neq 1$  are impractical, often leading to overflows, and to losses of accuracy for reasonably well-conditioned

Toeplitz systems. By extensive testing [KO97b] it was found that fast algorithms corresponding to the choice  $\phi = 1$  allow good numerical properties.

However the choice  $|\phi| = 1$  means that the displacement operator on (7.7) has a non-trivial kernel [consisting of all  $\phi$ -circulant matrices], so  $R$  is a partially reconstructible Toeplitz-like matrix.

In Sec. 5.3 we specified fast GE with diagonal pivoting algorithms for partially reconstructible Cauchy-like matrices. So the only point remaining to clarify is how to efficiently organize the transformation to a Cauchy-like matrix for partially reconstructible Toeplitz-like matrices.

**7.4. Transformation of partially reconstructible Toeplitz-like matrices to partially reconstructible Cauchy-like matrices.** Here we shall specify the definition of Sec. 5.1 of a generator for a partially reconstructible Toeplitz-like matrix  $R$  in

$$(7.9) \quad \nabla_{Z_1}(R) = R - Z_1 \cdot R \cdot Z_1^*.$$

As is well known [see, e.g., [GO92]], the kernel of  $\nabla_{Z_1}(\cdot)$  in (7.9) is the subspace of all circulants, i.e.,

$$(7.10) \quad \mathcal{C} = \{\text{Circ}(c) : c \in \mathbf{C}^n\} \subset \mathbf{C}^{n \times n}$$

where

$$\text{Circ}(c) = \begin{bmatrix} c_0 & c_{n-1} & \cdots & c_2 & c_1 \\ c_1 & c_0 & c_{n-1} & \cdots & c_2 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ c_{n-2} & & c_1 & c_0 & c_{n-1} \\ c_{n-1} & c_{n-2} & \cdots & c_1 & c_0 \end{bmatrix}.$$

Given arbitrary  $R$ , the next lemma and corollary show how to compute in  $O(n^2)$  arithmetic operations the decomposition of  $R$  with respect to  $\mathbf{C}^{n \times n} = \mathcal{C} \oplus \mathcal{C}^\perp$ .

**LEMMA 7.1.** *Let  $\mathcal{C} \subset \mathbf{C}^{n \times n}$  denote the subspace of all circulant matrices. use an inner product in  $\mathbf{C}^{n \times n}$  defined by (5.3), and let  $R = [r_{ij}] \in \mathbf{C}^{n \times n}$  be arbitrary. Then  $R \perp \mathcal{C}$  if and only if for  $k = 1, 2, \dots, n$  we have  $\sum_{i=1}^n r_{i+k-1,i} = 0$  [Here the indices are integers modulo  $n$ , i.e.. if  $k > n$ . then  $k := k - n$ ].*

Briefly, a matrix  $R$  is orthogonal [in the sense of (5.3)] to all circulants if and only if for each of its circulant diagonals, the sum of the entries is zero.

**Proof.** It is easy to check that  $\langle Z_1^k, R \rangle = \sum_{i=1}^n r_{i+k-1,i}$ . Since  $\{I, Z_1, Z_1^2, \dots, Z_1^{n-1}\}$  form a basis in  $\mathcal{C} \subset \mathbf{C}^{n \times n}$ , the assertions follow.  $\diamond$

**COROLLARY 7.2.** *With  $\mathcal{C} \subset \mathbf{C}^{n \times n}$  as above, a matrix  $R = [r_{ij}] \in \mathbf{C}^{n \times n}$  is decomposed with respect to  $\mathbf{C}^{n \times n} = \mathcal{C} \oplus \mathcal{C}^\perp$  as*

$$(7.11) \quad R = R_{\mathcal{C}} + R_{\mathcal{C}^\perp},$$

with

$$(7.12) \quad R_{\mathcal{C}} = \text{Circ}(c_0, c_1, \dots, c_{n-1}), \quad \text{where} \quad c_{i-1} = \frac{1}{n} \sum_{k=1}^n r_{i+k-1,k}.$$

and the indices are again integers modulo  $n$ .

For example, the ordinary Hermitian Toeplitz matrix has a  $\nabla_{Z_1}$ -generator

$$\left\{ \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} \\ t_1 - t_{n-1}^* & t_1 - t_{n-1}^* \\ \vdots & \vdots \\ t_{n-1} - t_1^* & t_{n-1} - t_1^* \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \text{Circ}(c_0, c_1, \dots, c_{n-1}) \right\},$$

where  $c_i = \frac{(n-i) \cdot t_i + i \cdot t_{i-n}}{n}$ . To transform  $R$  to a partially reconstructible Cauchy-like matrix we specify (7.8),

$$(7.13) \quad Z_1 = \mathcal{F}^* \cdot D_1 \cdot \mathcal{F}, \quad \text{with} \quad D_1 = \text{diag}(1, w, w^2, \dots, w^{n-1}),$$

where  $\omega = e^{\frac{2\pi i}{n}}$  is the  $n$ -th primitive root of unity. It is a trivial exercise to check that the kernel of  $\nabla_{D_1}(R) = R - D_1 R D_1^*$  is the subspace

$$\mathcal{D} = \{\text{diag}(f) : f \in \mathbf{C}^n\} \subset \mathbf{C}^{n \times n}$$

of all diagonal matrices. Now we are ready to state the transformation result.

**LEMMA 7.3.** (*Discrete-time transformation*) Let  $Z_1, D_1$  be as in (7.13), and let  $R = [r_{ij}]$  be a partially reconstructible Toeplitz-like matrix, given by a  $\nabla_{Z_1}$ -generator  $\{G_{\nabla_{Z_1}}, J, \text{Circ}(c)\}$ , i.e.,

$$\nabla_{Z_1}(R) = R - Z_1 \cdot R \cdot Z_1^* = G_{\nabla_{Z_1}} \cdot J \cdot G_{\nabla_{Z_1}}^*,$$

and  $c_i = \frac{1}{n} \sum_{k=1}^n r_{i+k-1}$ . Then  $\mathcal{F} \cdot R \cdot \mathcal{F}^*$  is a partially reconstructible Cauchy-like matrix, satisfying

$$\nabla_{D_1}(\mathcal{F} \cdot R \cdot \mathcal{F}^*) = (\mathcal{F} \cdot R \cdot \mathcal{F}^*) - D_1 \cdot (\mathcal{F} \cdot R \cdot \mathcal{F}^*) \cdot D_1^* = G_{\nabla_{D_1}} \cdot J \cdot G_{\nabla_{D_1}}^*,$$

with a  $\nabla_{D_1}$ -generator  $\{G_{\nabla_{D_1}}, J, \text{diag}(d)\}$ , where

$$(7.14) \quad G_{\nabla_{D_1}} = \mathcal{F} \cdot G_{\nabla_{Z_1}}, \quad \text{and} \quad d = \sqrt{n} \cdot \mathcal{F} \cdot c.$$

◇

Lemma 7.3 naturally leads to the symmetry-preserving transformation-and-pivoting algorithm based on the discrete-time algorithm for Cauchy-like matrices formulated in Sec. 5.5. This algorithm is about twice as fast as the GKO algorithm [based on partial pivoting] formulated in Sec. 3.5.

In the next lemma we specify a continuous-time analog of this transformation which is immediately deduced using the simple technique of Sec. 5.4.

**LEMMA 7.4.** (*Continuous-time transformation*) Let  $R$  be a partially reconstructible Toeplitz matrix, satisfying

$$(7.15) \quad \nabla_{Z_1}(R) = R - Z_1 \cdot R \cdot Z_1^* = G_{\nabla_{Z_1}} \cdot J \cdot G_{\nabla_{Z_1}}^*,$$

so that it is given by a  $\nabla_{Z_1}$ -generator  $\{G_{\nabla_{Z_1}}, J, \text{Circ}(c)\}$ . Define

$$A_1 = \text{diag}\left(\frac{\tau+1}{\tau-1}, \frac{\tau+w}{\tau-w}, \frac{\tau+w^2}{\tau-w^2}, \dots, \frac{\tau+w^{n-1}}{\tau-w^{n-1}}\right), \quad \text{where} \quad w = e^{\frac{2\pi i}{n}},$$

with  $\tau$  be any number with  $|\tau| = 1$  so that  $\tau^n \neq 1$ . Then  $\mathcal{F} \cdot R \cdot \mathcal{F}^*$  is a Cauchy-like matrix, satisfying, , satisfying

$$(7.16) \quad \nabla_{A_1}(\mathcal{F} \cdot R \cdot \mathcal{F}^*) = A_1 \cdot (\mathcal{F} \cdot R \cdot \mathcal{F}^*) + (\mathcal{F} \cdot R \cdot \mathcal{F}^*) \cdot A_1^* = G_{\nabla_{A_1}} \cdot J \cdot G_{\nabla_{A_1}}^*,$$

with a  $\nabla_{A_1}$ -generator  $\{G_{\nabla_{A_1}}, J, \text{diag}(d)\}$ , where

$$(7.17) \quad G_{\nabla_{A_1}} = \sqrt{2} \cdot (\tau I - D_1) \cdot \mathcal{F} \cdot G_{\nabla_{Z_1}}, \quad \text{and} \quad d = \sqrt{n} \cdot \mathcal{F} \cdot c.$$

slow	slow	fast	fast	fast	fast
GEPP	SGEBKP	GKO [partial] [pivoting]	cont-TP without diagonal pivoting	disc-TP without diagonal pivoting	
7.8e-06	1.6e-05	2.5e-05	1.3e+00	2.2e-06	6.1e-01
					4.5e-06

TABLE 7. *Chebyshev Toeplitz matrix with  $a = 0.2$ .*

This lemma naturally leads to one more symmetry-preserving transformation-and-pivoting algorithm based on the continuous-time algorithm for Cauchy-like matrices formulated in Sec. 5.3. This algorithm is also about twice as fast as GKO algorithm formulated in Sec. 3.5.

**7.5. Numerical example.** We refer to [KO97b] for a thorough comparison of the numerical performance of both discrete-time and continuous-time versions of fast GE with diagonal pivoting with several other available algorithms, including the usual slow GEPP, the fast  $O(n^2)$  classical Schur and Levinson algorithms, etc. Here we just present two illustrations suggesting that the approach indeed works numerically. In Table 7 we include the forward relative error for a linear system with a  $70 \times 70$  Chebyshev-Toeplitz coefficient matrix, i.e., with an indefinite symmetric Toeplitz matrix with the first row  $[ T_0(a) \dots T_{35}(a) \ 0 \dots 0 ]$  [tabulation of Chebyshev polynomials at a certain point  $a$ ]. We note that the three-term recurrence relations  $xT_k(x) = \frac{T_{k-1}(x) + T_{k+1}(x)}{2}$  for Chebyshev polynomials imply that all  $k \times k$  leading minors are singular for  $k = 3, 4, \dots, 35$ . This means that the classical fast Schur and Levinson Toeplitz solvers will just break down. The algorithms for the slow GEPP and SGEBKP [symmetric Gaussian elimination with Bunch-Kaufman pivoting] are taken from LAPACK, the GKO is the fast algorithm of Sec. 3.5, 4.5 [partial pivoting], cont-TP and disc-TP are the continuous-time and discrete-time transformation+pivoting algorithms, resp. We used the following variant of diagonal pivoting: first perform the symmetric pivoting step to maximize the (1,1) entry over the main diagonal, then perform the usual Bunch-Kaufman step. Table 7 illustrates the accuracy of fast pivoting techniques.

It is usually commented that symmetric pivoting with positive definite matrices is not necessary [because of the diagonal dominance], if one employs slow GE. However, many generally valid guidelines cannot be automatically carried over to problems where structured matrices and structure-exploiting algorithms are involved. To illustrate this point we included an example with [positive definite] ill-conditioned Gaussian Toeplitz matrix,  $T = [ a^{(i-j)^2} ]$ , arising, e.g., in image restoration problems. The next table shows the residual error for a  $70 \times 70$  Gaussian Toeplitz system. In this example we used symmetric pivoting [for diagonal pivoting], because for positive definite matrices it is equivalent to complete pivoting.

We refer to [GKO95] and [KO97b] for many other computed examples, and for practical recommendations, and here only briefly note that

- pivoting is needed to improve the accuracy of fast algorithms cont-TP and disc-TP even with positive definite matrices,

slow	GEPP		2.0e-07
slow	SGEBKP		1.4e-06
fast	Levinson		2.8e-04
fast	Schur		1.2e-07
fast	GKO	Partial pivoting	7.9e-07
fast	cont-TP	diagonal pivoting	1.9e-04
fast	cont-TP	no pivoting	1.6e-07
fast	disc-TP	diagonal pivoting	1.5e-03
fast	disc-TP	no pivoting	2.8e-07

TABLE 8. Gaussian Toeplitz matrix with  $a = 0.9$ .

- even with positive definite matrices the numerical properties of the Levinson algorithm are less favorable as compared to the Schur and other fast algorithms.
- Recall that for indefinite matrices the classical Schur and Levinson algorithms are inaccurate. Sometimes such inaccuracy can be overcomed via one step of iterative refinement, but there are examples, e.g., in Table 7, where there is nothing to refine, i.e., the classical Schur and Levinson algorithms just break down.

**7.6. Some remarks.** Toeplitz linear equations appear in numerous applications, and along with general algorithms, several fast  $O(n^2)$  solvers are available, e.g., the classical fast Schur and Levinson algorithms. These fast algorithms perform quite well for positive definite Toeplitz matrices [see, e.g. [GKO95] [KO97b] for numerical illustrations and recommendations, and [C80], [BBHS95] for the error analysis]. Nevertheless, their poor numerical properties for indefinite Toeplitz linear equations are well-known, the problem arises from the fact that the classical versions of the Levinson and the Schur algorithms recursively process all leading submatrices, they break down numerically if some of these submatrices are ill-conditioned. This fact motivated a number of the authors to design *look-ahead* versions for these fast algorithms; the idea behind the look-ahead approach is to jump over ill-conditioned submatrices which in practice has been shown to often improve accuracy [CH92], [FZ93], [GH94]. The language used in these and other look-ahead papers is rather different, varying from matrix arguments, through the use of the Padé table to applying a theory of formally biorthogonal polynomials. However, such look-ahead algorithms have their own limitations, for example, they are slow if the coefficient matrix has a large number of ill-conditioned submatrices [for example, for  $k = 3, 4, \dots, n/2$  the leading minors of the coefficient matrix in Table 7 are zero!]. So, the transformation-and-pivoting approach pursued in this section seems to be an attractive alternative, because it is simpler and it is always fast. We, however, have to point out that transformations of structured matrices is the known tool which has been used earlier by many authors for different purposes. An excellent survey paper [KN36] mentions that a Bezoutian [which is a Hankel-like matrix with displ. rank = 2] can be transformed into a Cauchy-like matrix and points out that this fact was known to Hermite. Same result can be found in a more explicit form in [L74]. Further, in [F84] it was shown how to transform a Hankel matrix into a Loewner matrix of the form  $\begin{bmatrix} & \\ & \frac{a_i - b_j}{x_i - y_j} \end{bmatrix}$  [which is, of course, a

Cauchy-like matrix], and a similar approach can be found in [L74]. For Toeplitz matrices transformations based on the discrete cosine/sine transforms were studied in [O93], [O95], however, in these papers the structure of the transformed matrix was not identified as a Cauchy-like one.

These results concern with the ordinary Toeplitz and Hankel matrices, and in [DGK81] it was suggested to replace the classical Schur-Cohn stability test by another method based on the classical Nevanlinna algorithm. A matrix interpretation of this method is the transformation of the Schur-Cohn matrix [which is Toeplitz-like, with displ. rank = 2] to a Pick matrix [which is Cauchy-like, with displ. rank = 2]. Interestingly, the motivation of [DGK81] was numerical: to reduce the stability test to a possibly better conditioned problem. However, since the transformation is based on unitary discrete transform matrices, such a reduction cannot lead to a better conditioned matrix. Furthermore, it [P90] [see also [BP94]] comments that transformations of Toeplitz-like, Vandermonde-like and Cauchy-like matrices with arbitrary displacement ranks can be potentially useful since they allow one to use an algorithm designed for one class of structure to solve problems for another. This idea was simplified and used in [GO94c] to design new algorithms for matrix-vector multiplication for Vandermonde-like and Cauchy-like matrices [via their transformation to Toeplitz-like matrices].

The judicious idea to combine the transformation technique with the possibility to pivot on Cauchy-like matrices was suggested in [H95], however the fast algorithm of [H95] is of a hybrid Levinson-Schur-type, and that algorithm, as it was stated, did not provide a satisfactory accuracy. In [GKO95] a variant of the fast transformation-and-pivoting algorithm was suggested as a combination of the results of [GO94c] [transformation to a Cauchy-like matrix], and of the fast GEPP of [GO93], [GO94b] [used to used to compute the triangular factorization for the transformed Cauchy-like matrix]. A large number of numerical experiments in [GKO95] and [KO97b] indicated that the approach is indeed practical, and that the algorithms of this type can be recommended as accurate in practice Toeplitz solvers. Further applications and results in this direction can be found in [KO95], [GTVDV96], [KVB96], [KO97a], [HB97], [KL96], [K97].

An a-priori rounding error analysis for the GKO algorithm of [GKO95] has been performed in [SB95], yielding weak stability. Along with the usual *element-growth-factor* [appearing also in the error bounds of the usual structure-ignoring GE], the corresponding error bound for the GKO algorithm also involves one more term, called a *generator-growth-factor*. This bound suggests that the accuracy of the usual slow GEPP and its fast implementation can be different. More precisely, if the element-growth-factor is small, but the generator-growth-factor is large, then the accuracy of the fast GKO algorithm can be less than the one of the slow GEPP. This bound motivated further modifications of the GKO algorithm, based on further fast implementations of several improved pivoting techniques [G95], [S96] aimed at the reduction of the generator-growth-factor.

All these results are quite recent, and, of course, only preliminary conclusions can be made at the moment. At the same time our current numerical experience [and the private communications with R.Brent, D.Sweet, M.Gu and M.Stewart] suggests that although there constructed few examples of Toeplitz matrices with larger generator-growth-factor [SB95], [S96], nevertheless the size of this factor is typically almost invariable comparable with the size of the element-growth factor.

and the GKO algorithm [i.e., with partial pivoting] seems to be quite reliable in practice. To sum up, we think that the comments made and cited in Sec. 2.2 for GE and unstructured matrices can be applied to structured matrices and fast algorithms as well.

### 7.7. Transformations for polynomial-Hankel-like matrices.

7.7.1. *Polynomial Hankel-like matrices.* To motivate introducing a general class of polynomial-Hankel-like matrices let us look first more closely at its important special case of Toeplitz-plus-Hankel-like matrices. Consider the displacement equation

$$(7.18) \quad \nabla_{H_Q}(R) = H_Q^T \cdot R - R \cdot H_Q = GJG^T,$$

with a tridiagonal matrix

$$(7.19) \quad H_Q = \frac{1}{2} \text{tridiag} \begin{bmatrix} * & 1 & 1 & \cdots & 1 & * \\ * & 0 & 0 & \cdots & 0 & 0 & * \\ * & 1 & 1 & \cdots & 1 & * \end{bmatrix}$$

where we do not specify the entries denoted by \*, to exploit this freedom later. It is well-known that the  $\nabla_{H_Q}$ -displacement rank of a sum  $R = T + H$  of any Toeplitz and Hankel matrices does not exceed four, see, e.g., table 4 for references [reader can easily verify this fact by exploiting shift-invariances in  $R = T + H$  along with the observation that  $H_Q$  is essentially a combination of lower and upper shifts]. A general transformation-to-Cauchy-and-pivoting scheme of Sec. 7.1 can be applied to Toeplitz-plus-Hankel-like matrices as well [GKO95], and it is based on the use of real discrete cosine/sine transforms [as opposed to the complex FFT used in Sec. 7.2].

We shall return to Toeplitz-plus-Hankel-like matrices in Sec. 7.7.4 after clarifying such transformation for the more general *polynomial-Hankel-like matrices* [KO96]. These matrices are defined via (7.18), but with a general upper Hessenberg matrix

$$(7.20) \quad H_Q = \begin{bmatrix} a_{01} & a_{02} & \cdots & \cdots & a_{0,n} \\ a_{11} & a_{12} & \cdots & \cdots & a_{1,n} \\ 0 & a_{22} & \cdots & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & a_{n-1,n-1} & a_{n-1,n} \end{bmatrix}$$

[of course, tridiagonal matrix (7.19) appears as a special case]. In [KO96] matrices having low  $\nabla_{H_Q}$ -displacement rank have been *polynomial Hankel-like matrices*, because they can be seen as Hankel-like matrices related to polynomials  $Q = \{Q_0(x), Q_1(x), \dots, Q_n(x)\}$  defined by

$$(7.21) \quad x \cdot Q_{k-1}(x) = a_{k,k} \cdot Q_k(x) + a_{k-1,k} \cdot Q_{k-1}(x) + \dots + a_{0,k} \cdot Q_0(x)$$

[Many useful results on connection of Hessenberg matrices  $H_Q$  to polynomials  $Q$  can be found in [MB79] where the name *confederate matrix* was associated with  $H_Q$  in (7.20)].

7.7.2. *Generators for polynomial Hankel-like matrices.* Since (7.18) has a non-trivial kernel, such polynomail-Hankel-like matrices  $R$  are partially reconstructible. Therefore we need a description for the kernel of  $\nabla_{H_Q}$ , and it is provided by the next theorem.

**THEOREM 7.5.** *Let us choose and fix an arbitrary invertible diagonal matrix  $W_Q$ , and denote by*

$$(7.22) \quad V_Q = \begin{bmatrix} Q_0(x_1) & Q_1(x_1) & \cdots & Q_{n-1}(x_1) \\ Q_0(x_2) & Q_1(x_2) & \cdots & Q_{n-1}(x_2) \\ \vdots & \vdots & & \vdots \\ Q_0(x_n) & Q_1(x_n) & \cdots & Q_{n-1}(x_n) \end{bmatrix}$$

*a polynomial-Vandermonde matrix whose nodes  $\{x_k\}$  are the zeros of the  $n$ -th polynomial  $Q_n(x)$  in (7.21). Then the kernel of  $\nabla_{H_Q}(\cdot)$  in (7.18) has the form*

$$(7.23) \quad \mathcal{K} = \text{span}\{(H_Q^T)^k \cdot (V_Q^* W_Q^2 V_Q), \quad k = 0, 1, \dots, n-1\}.$$

*Furthermore, any  $R_K \in \mathcal{K} = \text{Ker } \nabla_{H_Q}$  can be represented as*

$$(7.24) \quad R_K = \sum_{k=0}^{n-1} r_k \cdot Q_k(H_Q^T) \cdot (V_Q^T W_Q^2 V_Q).$$

Thus, a partially reconstructible polynomial Hankel-like matrix  $R$  in (7.18) can be represented by its generator  $\{G, J, R_K\}$  where the last matrix  $R_R$  is described by only  $n$  parameters  $\{r_k\}$  in (7.24). One special case will be considered in Sec. 7.7.4, and here we shall indicate another important one. Consider the simplest case  $Q = \{\frac{1}{\sqrt{n}}, \frac{x}{\sqrt{n}}, \frac{x^2}{\sqrt{n}}, \dots, \frac{x^{n-1}}{\sqrt{n}}, \frac{x^n - 1}{\sqrt{n}}\}$ , then it is easy to see that  $H_Q = Z_1$ , the lower shift circulant matrix defined in (7.5). So, it is clear that the kernel of (7.18) are all circulants [cf. Sec. 7.4]. Moreover, in this case  $V_Q$  is just a DFT matrix  $\mathcal{F} = \frac{1}{\sqrt{n}}(\omega^{ij})_{i,j=0}^{n-1}$  [with  $\omega$  denoting the primitive  $n$ -th root of unity]. Chosing  $W_Q = I$ , we note that since  $\mathcal{F}$  is unitary and  $Q_k(H_Q) = \frac{1}{\sqrt{n}}Z_1^k$ , the equation (7.24) reduces to the obvious description of a circulant by the entries of its first column  $\{\frac{r_k}{\sqrt{n}}\}$ .

**7.7.3. Transformations to Cauchy-like matrices.** Now, our goal is to transform  $R$  in (7.18) to a Cauchy-like matrix, so we need to diagonalize  $H_Q$ . For this, an easily verified formula involving the *polynomial Vandermonde matrix*  $V_Q(x)$  defined in (7.22) is readily available:

$$(7.25) \quad H_Q = V_Q^{-1} D_x V_Q, \quad \text{with} \quad D_x = \text{diag}(x_1, x_2, \dots, x_n).$$

where  $\{x_k\}$  are the zeros of  $Q_n(x)$ . This identity (7.25) can be rewritten as  $H_Q = V_Q^{-1} W_Q^{-1} D_x W_Q V_Q$  [as above,  $W_Q$  is an arbitrary diagonal matrix] which leads to the following transformation result.

**THEOREM 7.6.** *Let  $R$  be a polynomial Hankel-like matrix in (7.18), given by its  $\nabla_{H_Q}$ -generator  $\{G, J, R_K\}$ , and  $W_Q$  denotes an arbitrary invertible diagonal matrix. Then  $W_Q^{-T} V_Q^{-T} R V_Q^{-1} W_Q^{-1}$  is a Cauchy-like matrix with respect to*

$$\nabla_{D_x}(R) = D_x R - R D_x \quad [\text{with } D_x \text{ is given by (7.25)}]$$

*with a  $\nabla_{D_x}$ -generator*

$$(7.26) \quad \{W_Q^{-T} V_Q^{-T} G, \quad J, \quad W_Q^{-T} V_Q^{-T} R_K V_Q^{-1} W_Q^{-1}\}.$$

*where*

$$W_Q^{-T} V_Q^{-T} R_K V_Q^{-1} W_Q^{-1} = \begin{bmatrix} r(x_1) & & & \\ & \ddots & & \\ & & \ddots & \\ & & & r(x_n) \end{bmatrix}$$

where the diagonal entries are computed via a polynomial Vandermonde transform

$$\begin{bmatrix} r(x_1) \\ \vdots \\ r(x_n) \end{bmatrix} = V_Q \begin{bmatrix} r_0 \\ \vdots \\ r_{n-1} \end{bmatrix}.$$

with  $\{r_k\}$  from (7.24).

Returning to our comment made after theorem 7.5 we may note that this theorem generalizes the well-known result on diagonalization of circulant matrices by the discrete Fourier transform. Another important special case is considered next.

#### 7.7.4. Toeplitz-plus-Hankel-like matrices. Discrete Cosine and Sine transforms.

Here we return to the special case (7.19) for the displacement operator (7.18) associated in Sec. 7.7.1 with Toeplitz-plus-Hankel matrices. Now, using definitions and notations of Sec. 7.7.1 we notice that the matrix  $H_Q$  in a (7.19) is a confederate matrix whose associated polynomial system (7.21) satisfies the three-term recurrence relations,

$$(7.27) \quad xQ_k(x) = \frac{Q_{k-1}(x) + Q_{k+1}(x)}{2}.$$

Of course, these are the well-known three-term recurrence relations for Chebyshev polynomials which are essentially cosines and sines:

$$(7.28) \quad T_k(x) = \cos(k \arccos x), \quad U_k = \frac{\sin((k+1) \arccos x)}{\sin(\arccos x)}.$$

But here we have a complete freedom to fill in asterisks in (7.19) which means the freedom in choosing the first and the last polynomials of  $Q$ . For our purposes here it is convenient to make the following 8 choices listed in the second column of Table 9. The point in making these choices is that for these tridiagonal matrices  $H_Q$  we are able to write down the explicit expressions for the corresponding scaled polynomial Vandermonde matrices,  $W_Q V_Q$ , so that these matrices are just 8 versions of discrete cosine and sine transforms listed in Table 10.

There are well-known superfast  $O(n \log n)$  and accurate real arithmetic algorithms for all 8 matrices  $W_Q V_Q$  listed in the second column of Table 10. Therefore we are able to specify the results of Sec. 7.7 and to suggest a variety of real-arithmetic DCT/DST-based alternatives to the complex-arithmetic FFT-based transformation-and-pivoting methods of Sec. 7.2, 7.4. More details can be found in [KO96], where this transformation-to-Cauchy-like approach was used to derive in a uniform manner 8 DCT/DST-based analogs of T.Chan circulant preconditioner for Toeplitz linear equations [the usual T.Chan preconditioner is a circulant matrix, i.e. it is associated with the complex-arithmetic DFT].

## 8. Ordinary Cauchy matrices. Specific algorithms and predictive pivoting

The algorithms in Sec. 3.5, 5.3 were designed for the general Cauchy-like matrices, some applications for this general case were considered in Sec. 4 and 6. However, a special case of ordinary Cauchy matrices  $C(x, y) = \left[ \frac{1}{x_i - y_j} \right]$  is also of interest in several areas, see, e.g., [B37], [C41], [Ro85], [T86], [OS86], [R90a],

DCT-I	$H_Q = \text{tridiag} \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} & \frac{1}{2} & \cdots & \frac{1}{2} & 0 & \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & 0 & 0 & \cdots & 0 & \frac{1}{2} & 0 & \frac{1}{\sqrt{2}} \\ & \frac{1}{2} & \frac{1}{2} & \cdots & & \frac{1}{2} & 0 & \frac{1}{\sqrt{2}} \end{bmatrix}$
DCT-II	$H_Q = \text{tridiag} \begin{bmatrix} \frac{1}{2} & 0 & \frac{1}{2} & \cdots & \frac{1}{2} & 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} & \cdots & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ & \frac{1}{2} & \frac{1}{2} & \cdots & & \frac{1}{2} & 0 & \frac{1}{2} \end{bmatrix}$
DCT-III	$H_Q = \text{tridiag} \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} & \frac{1}{2} & \cdots & \frac{1}{2} & 0 & \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & 0 & 0 & \cdots & 0 & \frac{1}{2} & 0 & \frac{1}{\sqrt{2}} \\ & \frac{1}{2} & \frac{1}{2} & \cdots & & \frac{1}{2} & 0 & \frac{1}{\sqrt{2}} \end{bmatrix}$
DCT-IV	$H_Q = \text{tridiag} \begin{bmatrix} \frac{1}{2} & 0 & \frac{1}{2} & \cdots & \frac{1}{2} & 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} & \cdots & 0 & \frac{1}{2} & 0 & -\frac{1}{2} \\ & \frac{1}{2} & \frac{1}{2} & \cdots & & \frac{1}{2} & \frac{1}{2} & \end{bmatrix}$
DST-I	$H_Q = \text{tridiag} \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & \cdots & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & \cdots & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ & \frac{1}{2} & \frac{1}{2} & \cdots & & \frac{1}{2} & 0 & \frac{1}{2} \end{bmatrix}$
DST-II	$H_Q = \text{tridiag} \begin{bmatrix} -\frac{1}{2} & 0 & \frac{1}{2} & \cdots & \frac{1}{2} & 0 & \frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} & \cdots & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ & \frac{1}{2} & \frac{1}{2} & \cdots & & \frac{1}{2} & \frac{1}{2} & \end{bmatrix}$
DST-III	$H_Q = \text{tridiag} \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & \cdots & \frac{1}{2} & 0 & \sqrt{\frac{1}{2}} & 0 \\ \frac{1}{2} & 0 & 0 & \cdots & 0 & \frac{1}{2} & 0 & \sqrt{\frac{1}{2}} \\ & \frac{1}{2} & \frac{1}{2} & \cdots & & \frac{1}{2} & \sqrt{\frac{1}{2}} & 0 \end{bmatrix}$
DST-IV	$H_Q = \text{tridiag} \begin{bmatrix} -\frac{1}{2} & 0 & \frac{1}{2} & \cdots & \frac{1}{2} & 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} & \cdots & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ & \frac{1}{2} & \frac{1}{2} & \cdots & & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix}$

TABLE 9. Choices of  $H_Q$  in (7.19).

[HLM95], [RS94], [MS77], [BGR90]. Recall that the ordinary Cauchy matrices have displacement rank one,

$$(8.1) \quad \nabla_{\{D_x, D_y\}}(C(x, y)) = D_x C(x, y) - C(x, y) D_y = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}.$$

	Discrete transform	Inverse transform
DCT-I	$C_N^I = \sqrt{\frac{2}{N-1}} \left[ \alpha_{kj} \cos \frac{kj\pi}{N-1} \right]_{k,j=0}^{N-1}$ where $\alpha_{kj} = \eta_k \eta_{N-1-k} \eta_j \eta_{N-1-j}$	$[C_N^I]^{-1} = [C_N^I]^T = C_N^I$
DCT-II	$C_N^{II} = \sqrt{\frac{2}{N}} \left[ \eta_k \cos \frac{k(2j+1)\pi}{2N} \right]_{k,j=0}^{N-1}$	$[C_N^{II}]^{-1} = [C_N^{II}]^T = C_N^{III}$
DCT-III	$C_N^{III} = \sqrt{\frac{2}{N}} \left[ \eta_j \cos \frac{(2k+1)j\pi}{2N} \right]_{k,j=0}^{N-1}$	$[C_N^{III}]^{-1} = [C_N^{III}]^T = C_N^{II}$
DCT-IV	$C_N^{IV} = \sqrt{\frac{2}{N}} \left[ \cos \frac{(2k+1)(2j+1)\pi}{4N} \right]_{k,j=0}^{N-1}$	$[C_N^{IV}]^{-1} = [C_N^{IV}]^T = C_N^{IV}$
DST-I	$S_N^I = \sqrt{\frac{2}{N+1}} \left[ \sin \frac{kj}{N+1}\pi \right]_{k,j=1}^N$	$[S_N^I]^{-1} = [S_N^I]^T = S_N^I$
DST-II	$S_N^{II} = \sqrt{\frac{2}{N}} \left[ \eta_k \sin \frac{k(2j-1)}{2N}\pi \right]_{k,j=1}^N$	$[S_N^{II}]^{-1} = [S_N^{II}]^T = S_N^{III}$
DST-III	$S_N^{III} = \sqrt{\frac{2}{N}} \left[ \eta_j \sin \frac{(2k-1)j}{2N}\pi \right]_{k,j=1}^N$	$[S_N^{III}]^{-1} = [S_N^{III}]^T = S_N^{II}$
DST-IV	$S_N^{IV} = \sqrt{\frac{2}{N}} \left[ \sin \frac{(2k-1)(2j-1)}{4N}\pi \right]_{k,j=1}^N$	$[S_N^{IV}]^{-1} = [S_N^{IV}]^T = S_N^{IV}$

TABLE 10. Discrete trigonometric transforms.

and this fact can be exploited to obtain more specific fast algorithms and more elaborated stabilizing techniques often yielding a remarkably high numerical accuracy. These results and their implications [e.g., to divided differences] are described in the rest of the paper.

**8.1. Direct generator recursion. GS-direct algorithm.** To begin with, it is possible to implement the generator recursion (3.5) for  $C(x, y)$  as a direct manipulation on the nodes  $\{x_k\}$ ,  $\{y_k\}$ , as shown in the next statement.

LEMMA 8.1. (*GS-direct algorithm*) Let  $R_k$  in

$$\text{diag}(x_k, \dots, x_n)R_k - R_k \text{diag}(y_k, \dots, y_n) = G_k B_k =$$

$$\begin{bmatrix} g_k^{(k)} & \dots & g_n^{(k)} \end{bmatrix}^T \begin{bmatrix} b_k^{(k)} & \dots & b_n^{(k)} \end{bmatrix}$$

be the successive Schur complements of the Cauchy matrix  $C(x, y) = \left[ \frac{1}{x_i - y_j} \right]$ . Then the generator recursion (3.8) can be specialized to

$$\begin{bmatrix} g_{k+1}^{(k+1)} \\ \vdots \\ g_n^{(k+1)} \end{bmatrix} = \begin{bmatrix} \frac{x_{k+1}-x_k}{x_{k+1}-y_k} g_{k+1}^{(k)} \\ \vdots \\ \frac{x_n-x_k}{x_n-y_k} g_n^{(k)} \end{bmatrix},$$

(8.2)

$$\begin{bmatrix} b_{k+1}^{(k+1)} & \dots & b_n^{(k+1)} \end{bmatrix} = \begin{bmatrix} \frac{y_{k+1}-y_k}{y_{k+1}-x_k} b_n^{(k)} & \dots & \frac{y_n-y_k}{y_n-x_k} b_n^{(k)} \end{bmatrix}.$$

The nonzero entries of the factors in  $C(x, y) = LDU$  are given by

$$(8.3) \quad \begin{aligned} \begin{bmatrix} l_{k,k} \\ \vdots \\ l_{n,k} \end{bmatrix} &= \begin{bmatrix} \frac{1}{x_k - y_k} g_k^{(k)} \\ \vdots \\ \frac{1}{x_n - y_k} g_n^{(k)} \end{bmatrix}, \\ d_k &= x_k - y_k \\ \begin{bmatrix} u_{k,k} & \cdots & u_{k,n} \end{bmatrix} &= \begin{bmatrix} \frac{1}{x_k - y_k} b_k^{(k)} & \cdots & \frac{1}{x_k - y_n} b_n^{(k)} \end{bmatrix} \end{aligned}$$

A fast Cauchy solver based on recursion (8.2) (8.3) was called *GS-direct algorithm* in [BKO95b], and it requires only  $6n^2$  flops, but it needs  $O(n^2)$  locations of memory [to store the triangular factors of  $C(x, y)$ ].

**8.2. Quasi-Cauchy algorithm.** A more efficient fast algorithm requiring just  $O(n)$  memory locations is based on the next statement.

LEMMA 8.2. (*quasi-Cauchy algorithm*) *The Cauchy matrix and its inverse can be factored as*

$$(8.4) \quad C(x, y) = L_1 \dots L_{n-1} D U_{n-1} \dots U_1,$$

where  $D = \text{diag}((x_1 - y_1), \dots, (x_n - y_n))$ , and

$$\begin{aligned} L_k &= \left[ \begin{array}{c|ccccc} I_{k-1} & & & & & \\ \hline & \frac{1}{x_k - y_k} & & & & \\ & & \ddots & & & \\ & & & \frac{1}{x_n - x_k} & & \end{array} \right] \left[ \begin{array}{c|ccccc} I_{k-1} & & & & & \\ \hline & 1 & & & & \\ & 1 & 1 & & & \\ & \vdots & & \ddots & & \\ & 1 & & & & 1 \end{array} \right] \times \\ &\quad \left[ \begin{array}{c|ccccc} I_{k-1} & & & & & \\ \hline & 1 & & & & \\ & & x_{k+1} - x_k & & & \\ & & & \ddots & & \\ & & & & & x_n - x_k \end{array} \right], \\ U_k &= \left[ \begin{array}{c|ccccc} I_{k-1} & & & & & \\ \hline & 1 & & & & \\ & & y_k - y_{k+1} & & & \\ & & & \ddots & & \\ & & & & y_k - y_n & \end{array} \right] \left[ \begin{array}{c|ccccc} I_{k-1} & & & & & \\ \hline & 1 & 1 & \dots & 1 & \\ & & 1 & & & \\ & & & \ddots & & \\ & & & & & 1 \end{array} \right] \times \\ &\quad \left[ \begin{array}{c|ccccc} I_{k-1} & & & & & \\ \hline & \frac{1}{x_k - y_{k+1}} & & & & \\ & & \ddots & & & \\ & & & \frac{1}{x_k - y_n} & & \end{array} \right]. \end{aligned}$$

The factorization (8.4) is easily deduced from Lemma 8.1 by factoring the successive Schur complements as

$$R_k = L_k C(x, y) U_k$$

at each step of the recursion [Here we use the same designation  $C(x, y)$  for all  $(n+1-k) \times (n+1-k)$  Cauchy matrices defined by the nodes  $\{x_k, \dots, x_n\}$  and  $\{y_k, \dots, y_n\}$ ,  $(k = 1, \dots, n)$ ].

Each of the factors  $L_k, U_k$  can be easily inverted by just changing the sign for the off-diagonal entries, thus leading to a new Cauchy solver. This fast Cauchy solver requires only  $6n^2$  operations per system and only  $O(n)$  memory locations. Because of a certain analogy with quasi-Toeplitz matrices, it was called *quasi-Cauchy* algorithm in [BKO95b].

**8.3. Error analysis.** An experience shows that without stabilizing techniques the GS-direct algorithm [based on Lemmas 8.1] and of quasi-Cauchy algorithm [based on Lemma 8.2] are often less accurate than the more expensive structure-ignoring GEPP. A remedy stabilizing the numerical performance of these fast algorithms will be suggested in Sec. 8.4 and 8.5. To motivate and justify this remedy we next present the results the error analysis. Here we assume the standard model of the floating point arithmetic (2.3) with unit roundoff  $u$ .

**THEOREM 8.3. (Accuracy of the GS-direct algorithm)** *If the triangular factorization computed via the GS-direct algorithm [based on Lemma 8.1] is used to solve the associated linear system  $C(x, y)a = f$  then the computed solution  $\hat{a}$  solves a nearby system  $(C(x, y) + \Delta C)\hat{a} = f$  with a backward error*

$$(8.5) \quad |\Delta C| \leq ((10n - 2)u + O(u^2))|L||DU|,$$

and a residual error

$$(8.6) \quad |C(x, y)\hat{a} - f| \leq ((10n - 2)u + O(u^2))|L||DU||\hat{a}|,$$

**THEOREM 8.4. (Accuracy of quasi-Cauchy algorithm)** *The solution  $\hat{a}$  of a Cauchy linear system  $C(x, y)a = f$  computed via the quasi-Cauchy algorithm [based on Lemma 8.2] solves a nearby system  $(C(x, y) + \Delta C)\hat{a} = f$  with a backward error*

$$(8.7) \quad |\Delta C| \leq ((n^2 + 11n - 10)u + O(u^2))|L||DU|.$$

and a residual error

$$(8.8) \quad |C(x, y)\hat{a} - f| \leq ((n^2 + 11n - 10)u + O(u^2))|L||DU||\hat{a}|,$$

We refer to [BKO95b] for more error bounds and for complete proofs, and next we turn to the implications of these results to the enhancement of the numerical performance of these algorithms.

**8.4. Numerical example: pivoting and GS-direct-Cauchy and quasi-Cauchy algorithms.** The backward error bounds (8.5), (8.7) and the residual error bounds (8.6), (8.8) of the two new fast algorithms are similar to the analogous bounds (2.5) and (2.6) of the GE procedure. This observation indicates that various pivoting techniques, so successful for GE, will also stabilize the numerical performance of the GS-direct-Cauchy and quasi-Cauchy algorithms.

Recall that the entries of the first rows and columns are immediately available during the recursive GS-direct-Cauchy algorithm, see, e.g., (8.3). Therefore it is straightforward to incorporate pivoting into this algorithm, thus achieving its numerical accuracy. In contrast, the quasi-Cauchy algorithm is not recursive, so we exploit next a different approach to incorporate pivoting.

slow	fast	fast	fast
GEPP	inversion	GS-direct partial pivoting	quasi-Cauchy partial pivoting
1e-06	6e-03	7e-07	2e-04
			6e-07
			4e-4

TABLE 11. *Cauchy-Toeplitz matrix*  $[1/(1 - 0.3(i - j))]$ .

**8.5. Predictive pivoting.** We started the paper with an example of a Vandermonde matrix,  $V(x)$ , for which partial pivoting ordering of the rows can be achieved *in advance* by computing in  $O(n^2)$  operations the *Leja ordering* (1.2) of the nodes  $\{x_k\}$ . This possibility is quite useful, because it allows us to incorporate pivoting into *any* fast  $O(n^2)$  algorithm for  $V(x)$ , without slowing it down. Ordinary Vandermonde and ordinary Cauchy matrices have many similar properties, for example they both have displacement rank one. Therefore, a natural question is: can we design a similar fast  $O(n^2)$  algorithm to compute in advance a partial pivoting ordering of the rows for an ordinary Cauchy matrix? This would allow us to incorporate partial pivoting into any fast  $O(n^2)$  Cauchy solver, e.g., into quasi-Cauchy algorithm.

As was noted in Sec. 2.4, partial pivoting of the rows of  $R$  successively maximizes the determinants of its leading submatrices. Vandermonde and Cauchy matrices have two nice properties in common: first, their leading submatrices have the same pattern of structure; and secondly, there are well-known explicit formulas for their determinants in terms of the nodes, e.g.,

$$(8.9) \quad \det C(x, y) = \frac{\prod_{j>k} (x_j - x_k) \prod_{j<k} (y_j - y_k)}{\prod_{j,k} (x_j - y_k)} \quad 1 \leq j, k \leq n.$$

This observation was used in Sec. 2.4 to express partial pivoting of the rows for  $V(x)$  as Leja ordering (1.2). Analogously, partial pivoting for  $C(x, y)$  can be obtained as a successive maximization of the quantities

$$(8.10) \quad |d_i| = \left| \frac{\prod_{j=1}^{i-1} (x_i - x_j) \prod_{j=1}^{i-1} (y_i - y_j)}{(x_i - y_i) \prod_{j=1}^{i-1} (x_i - y_j) \prod_{j=1}^{i-1} (x_j - y_i)} \right|, \quad (i = 1, \dots, n).$$

This procedure has been called *predictive partial pivoting* [BKO95a], because it can be rapidly computed *in advance* in only  $O(n^2)$  flops.

Predictive partial pivoting stabilizes in practice the numerical behavior of the GS-direct and quasi-Cauchy algorithms, as illustrated in Table 8.5 containing the backward error for a  $100 \times 100$  Cauchy-Toeplitz matrix  $\left[ \frac{1}{1-0.3(i-j)} \right]$ .

Note that the approach allows us to incorporate other pivoting techniques, e.g., Gu's pivoting, see, e.g., [BKO95b] for comparison and references.

**8.6. Totally positive matrices. Avoidance of pivoting.** Let us now again return to the example of  $V(x)$  given in section 1, and recall that for totally positive Vandermonde matrices,

$$(8.11) \quad 0 < x_1 < x_2 < \dots < x_n,$$

it is advantageous not to pivot when using structure-ignoring Gaussian elimination, see, e.g., the discussion at the end of Sec. 1 and the error bound (2.8) in Sec. 2.3.

A natural question would be: Is there an analogue of these results for our fast algorithms for Cauchy matrices  $C(x, y)$ ? We next present an affirmative answer to this question.

It is known [GK50] that the condition

$$(8.12) \quad y_n < \dots < y_1 < x_1 < \dots < x_n$$

is equivalent to the total positivity of  $C(x, y)$ . Under this condition the results of Theorems 8.3 and 8.4 imply that not only slow GE, but also the new fast GS-direct-Cauchy and quasi-Cauchy algorithms compute solutions with a favorable small backward error.

**THEOREM 8.5.** (*Backward stability of GS-direct algorithm for totally positive matrices*) Assume that condition (8.12) holds, i.e. that  $C(x, y)$  is totally positive. If the triangular factorization of the GS-direct algorithm is used to solve the associated linear system, then the computed solution  $\hat{a}$  solves a nearby system

$$(C(x, y) + \Delta C)\hat{a} = f,$$

with

$$(8.13) \quad |\Delta C| \leq ((10n - 3)u + O(u^2))C(x, y).$$

The analogous backward bound for the quasi-Cauchy algorithm is

$$(8.14) \quad |\Delta C| \leq ((n^2 + 11n - 10)u + O(u^2))C(x, y).$$

The above results show that the backward stability of the fast GS-direct and quasi-Cauchy algorithms for totally positive Cauchy matrices is even more favorable than that of the slow Gaussian elimination procedure. Indeed the difference is that for the GE procedure the bound (2.8) is valid only for the case when the entries of the *computed factors*  $\hat{L}$  and  $\hat{U}$  remain positive (which is usually not the case with ill-conditioned matrices), whereas the favorable bounds in the above theorem hold as long as there are no overflows. For example, for the Hilbert matrix  $H = [\frac{1}{i+j-1}]$  the condition number  $k_2(H)$  grows exponentially with the size, so already for small  $n$  we have  $k_2(H) > q(n)\frac{1}{u}$ . Here  $q(n)$  is a polynomial of small degree in  $n$ . Then in accordance with [W68] the matrix  $H$  will likely lose during the elimination not only its total positivity, but also the weaker property of being positive definite. Correspondingly, the single precision LAPACK routine **SPOSV** for Cholesky factorization, when applied to the Hilbert matrix, exits with an error flag already for  $n = 9$ , warning that the entries of  $\hat{L}$ ,  $\hat{U}$  became negative, so the pleasing backward bound (2.8) is no longer valid for Gaussian elimination. In contrast, the favorable bounds (8.13), (8.14) are valid for higher sizes, as long as there are no overflows.

For totally positive Cauchy matrices the GS-direct and the quasi-Cauchy algorithms have a remarkably small backward error, but their *forward accuracy* is not as high as the one of another new Cauchy solver, described in the next section.

## 9. Totally positive structured matrices and avoidance of pivoting: Björck-Pereyra-type algorithms

### 9.1. The Björck-Pereyra (BP) algorithm for Vandermonde matrices.

We again return to the example of a Vandermonde matrix of Sec. 1, and now discuss

the Björck-Pereyra (BP) algorithm [BP70]. This algorithm is based on the explicit decomposition of the inverse of a Vandermonde matrix into the product

$$(9.1) \quad V(x)^{-1} = U_1 \cdot \dots \cdot U_{n-1} \cdot L_{n-1} \cdot \dots \cdot L_1,$$

of the upper and lower *bidiagonal* factors [the entries of  $L_k, U_k$  are specified in [BP70] explicitly, without computation]. The BP algorithm solves the associated linear system by multiplying the representation (9.1) by a right-hand-side, and the sparse bidiagonal structure of the factors  $L_k, U_k$  yields a favorable efficiency of  $\frac{5}{2}n^2$  operations per system. This algorithm is now a well-known example where the exploitation of the structure allows one not only to speed-up computations, but also to achieve, for special right-hand sides, more accuracy in the computed solution than when using slow structure-ignoring methods. The first indication of this interesting phenomena can be found in [BP70], where it was observed that “*some problems, connected with Vandermonde systems, which traditionally have been considered to be too ill-conditioned to be attacked, actually can be solved with good precision*”.

This fact attracted much attention and motivated a number of papers appearing in the last decade. Specifically, many efforts were devoted to the extension of the Björck-Pereyra algorithm to more general classes of matrices. Thus, W.Tang and G.Golub [TG81] devised a closely related algorithm for block Vandermonde matrices, N.J.Higham extended in [Hig88] this algorithm to the so-called Vandermonde-like matrices involving orthogonal polynomials<sup>7</sup>, and L.Reichel and G.Opfer [RO91] specified a progressive [i.e., allowing updating] Björck-Pereyra-type algorithm for Chebyshev-Vandermonde matrices.

Another challenging problem was to give a theoretical support for the favorable numerical properties of the Björck-Pereyra algorithm by performing an *a priori* rounding error analysis, the results in this direction are given next.

- **Forward error bound.** It was shown in [Hig87] that for totally positive Vandermonde systems  $V(x)a = f$  the forward error of the BP algorithm is nicely bounded:

$$(9.2) \quad |a - \hat{a}| \leq 5nu|V(x)^{-1}||f| + O(u^2).$$

- **Backward error bound.** In [BKO95a] it was shown that for totally positive Vandermonde matrices not only forward, but also backward, error is remarkable small:

$$(9.3) \quad |\Delta V| \leq 12n^2uV(x)|\hat{a}| + O(u^2).$$

Here we may recall Wilkinson’s [W71] advocation not to give “... *much importance to the precise error bounds obtained by a priori error analysis*,” even they are often impractically large, they can “*expose the potential instabilities, if any, of an algorithm, so that hopefully from the insight thus obtained one might be led to improved algorithms*.” In contrast to many such impractical (though still useful) bounds, the bounds in (9.2), (9.3) are surprisingly favorable, predicting that the Björck-Pereyra algorithm can produce all 7 possible-in-single-precision digits, even in situations where Gaussian elimination with complete pivoting will fail to produce

---

<sup>7</sup>We call them three-term Vandermonde matrices, because we use the postfix “*like*” in a different meaning, which it has in the context of displacement structure theory.

as much as one correct digit. For example, the total positivity implies that for the sign-oscillating right-hand-side we have  $|V(x)^{-1}|f| = |V(x)^{-1}f|$ , so (9.2) becomes

$$(9.4) \quad |a - \hat{a}| \leq 5nu|a| + O(u^2),$$

see, e.g., [Hig87], implying the full relative accuracy.

**9.2. The BP-type algorithm for Cauchy matrices and numerical example.** As we already several times exploited an analogy between ordinary Vandermonde and Cauchy matrices, we naturally arrive at the following question: is there a counterpart of the BP algorithm for Cauchy matrices, and if yes, does it have a similar remarkable high forward and backward numerical accuracy? In this section we present affirmative answers to these questions. First we formulate the analog of the decomposition (9.1) into the product of *bidiagonal* factors.

LEMMA 9.1. *The inverse of the Cauchy matrix  $C(x, y)$  can be decomposed as*

$$(9.5) \quad C^{-1}(x, y) = U_1 U_2 \dots U_{n-1} D L_{n-1} \dots L_2 L_1,$$

where

$$(9.6) \quad L_k = \left[ \begin{array}{c|ccccc} I_k & & & & & \\ \hline & \frac{1}{x_{k+1}-x_1} & & & & \\ & & \frac{1}{x_{k+2}-x_2} & & & \\ & & & \ddots & & \\ & & & & \frac{1}{x_n-x_{n-k}} & \end{array} \right] \times$$

$$\left[ \begin{array}{c|ccccc} I_{k-1} & & & & & \\ \hline & 1 & 0 & & & \\ & -(x_1-y_k) & (x_{k+1}-y_k) & & & \\ & & & \ddots & & \\ & & & & -(x_{n-k}-y_k) & (x_n-y_k) \end{array} \right],$$

$$(9.7) \quad U_k = \left[ \begin{array}{c|ccccc} I_{k-1} & & & & & \\ \hline & 1 & -(x_k-y_1) & & & \\ & 0 & (x_k-y_{k+1}) & \ddots & & \\ & & & \ddots & -(x_k-y_{n-k}) & \\ & & & & (x_k-y_n) & \end{array} \right] \times$$

$$\left[ \begin{array}{c|ccccc} I_i & & & & & \\ \hline & \frac{1}{y_1-y_{k+1}} & & & & \\ & & \frac{1}{y_2-y_{k+2}} & & & \\ & & & \ddots & & \\ & & & & \frac{1}{y_{n-k}-y_n} & \end{array} \right]$$

$$(9.8) \quad D = \text{diag}\{(x_1 - y_1), (x_2 - y_2), \dots, (x_n - y_n)\}.$$

Using this representation the associated linear system can be solved in  $7n^2$  operations.

An error analysis of [BKO95a] produced the following counterparts of the bounds (9.2) and (9.3):

**THEOREM 9.2.** Suppose that the condition  $y_n < \dots < y_1 < x_1 < \dots < x_n$  holds [so that  $C(x, y)$  is totally positive, and the BP-type algorithm is carried out in floating-point arithmetic with unit round-off  $u$ , and that no overflows were encountered during computation]. Then the computed solution  $\hat{a}$  to  $C(x, y)a = f$  satisfies

(1) **Forward bound**

$$(9.9) \quad |a - \hat{a}| \leq 5(2n + 1) \cdot u \cdot |C(x, y)^{-1}| |f| + \mathcal{O}(u^2).$$

(2) **Backward bound** The computed solution  $\hat{a}$  is an exact solution of a nearby system  $\hat{C}\hat{a} = f$  with

$$(9.10) \quad |C(x, y) - \hat{C}| \leq c_n \cdot u \cdot C(x, y) + \mathcal{O}(u^2).$$

where  $c_n = 20n(2n - 1)$ .

(3) **Residual bound**

$$(9.11) \quad |f - C(x, y)\hat{a}| \leq c_n \cdot u \cdot C(x, y)|\hat{a}| + \mathcal{O}(u^2).$$

For example, similarly to (9.4) we have that for the sign-oscillating right-hand side we have

$$(9.12) \quad |a - \hat{a}| \leq 5(2n + 1)u|a| + O(u^2).$$

These results predict a remarkable small errors in the solution, computed via the BP-type algorithm, as illustrated in Figure 1 and Table 12 next example.

$n$	10	20	30	40	50	60
$\kappa_\infty(C(x, y))$	2e+08	4e+15	4e+21	4e+22	4e+22	7e+22

TABLE 12. Conditioning of Cauchy matrix with  $x_i = i^4/n^4$ .  $y_i = -i^4/n^4$ .

We see that the GE with complete pivoting fails to provide even one correct digit in the computed solution, which is expectable, given extremely large condition number. Nevertheless, the BP-type algorithm with monotonically ordered nodes (8.12) [thus preserving total positivity] gives us the full relative accuracy [i.e., about 7 digits out of about 7 possible in single precision].

**9.3. Effective stability and numerical example.** In [CF88] Chan and Foulser, motivated by the high accuracy of the BP algorithm for sign-oscillating right-hand side [see, e.g., (9.4)], and by some other examples, introduced the concept of *effective well-conditioning*. The latter reflects the fact that the sensitivity to perturbations of a solution depends upon the direction of the right-hand side vector, which suggests the potential existence of algorithms that can exploit the effective well-conditioning to produce higher accuracy for special right-hand sides. Note, however, that in general the numerical performance of a certain algorithm and the effective well-conditioning of a system to be solved have nothing to do with each other, and many algorithms are insensitive to the direction of the right-hand side vector. For example, for Gaussian elimination this point is illustrated by the numerical examples below. Here we use the approach of [CF88] to show that our new BP-type Cauchy solver is indeed an example of an algorithm that is guaranteed to accurately solve effectively well-conditioned Cauchy systems with totally positive coefficient matrices.

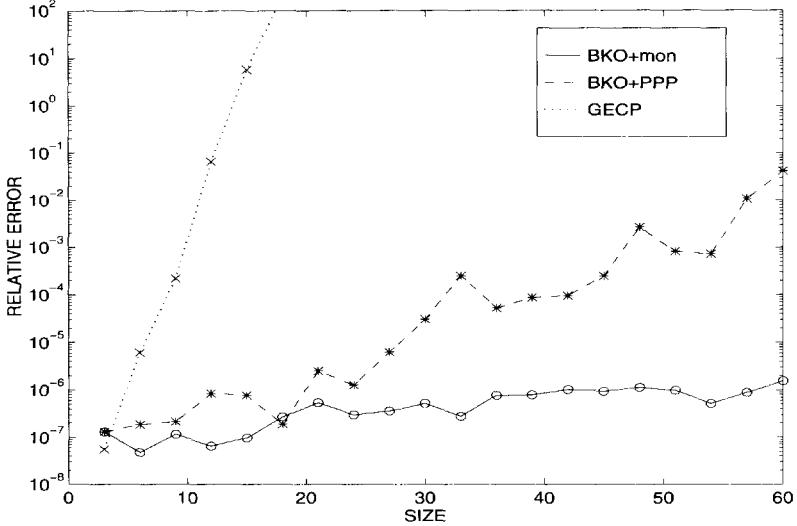


FIGURE 1. Cauchy linear system with  $x_i = i^4/n^4$ ,  $y_i = -i^4/n^4$ , and  $f_i = (-1)^i$ , ( $i = 1, \dots, n$ ). The graphs display the relative error  $\|a - \hat{a}\|_\infty/\|a\|_\infty$  as a function of  $n$ , for the following three algorithms : **BKO+mon**: BKO algorithm with monotonically ordered nodes, **BKO+PPP**: BKO algorithm with predictive partial pivoting, **GECP**: Gaussian elimination with complete pivoting.

Before stating the result, let us introduce the necessary notations. Let

$$(9.13) \quad \kappa_2(A, f) = \lim_{\varepsilon \rightarrow 0} \sup_{\|\Delta f\|_2 \leq \varepsilon \|f\|_2} \frac{\|\Delta a\|_2}{\|a\|_2 \cdot \varepsilon},$$

measures, in the 2-norm, the sensitivity of the solution of  $Aa = f$  to small perturbations in the right hand-side. Chan and Foulser derived an upper bound for  $\kappa_2(A, f)$  in terms of the direction of the right-hand side vector relative to the left singular vectors of  $A$ . To be more precise, let  $A = U \cdot \Sigma \cdot V^T$  be the singular value decomposition, where the columns of  $U = [u_1 \ u_2 \ \dots \ u_n]$  are the left singular vectors, the columns of  $V = [v_1 \ v_2 \ \dots \ v_n]$  are the right singular vectors, and the diagonal entries of  $\Sigma = \text{diag}\{\sigma_1 \ \sigma_2 \ \dots \ \sigma_n\}$  are the singular values of  $A$  in decreasing order. Denote by  $P_k = [u_{n+1-k} \ \dots \ u_n] \cdot [u_{n+1-k} \ \dots \ u_n]^T$  the projection operator onto the linear span of the smallest  $k$  left singular vectors of  $A$ . Then, in accordance with [CF88],

$$(9.14) \quad \kappa_2(A, f) \leq \gamma(A, f) \leq \kappa_2(A),$$

where

$$(9.15) \quad \gamma(A, f) = \min_k \frac{\sigma_{n-k+1}}{\sigma_n} \cdot \frac{\|f\|_2}{\|P_k f\|_2},$$

and  $\kappa_2(A) = \|A^{-1}\|_2 \cdot \|A\|_2$  is the 2-norm condition number. Note that the second inequality in (9.14) can easily be deduced by inspecting (9.15) for  $k = n$ . At the same time, if a large part of  $f$  lies in the span of the small left singular vectors of

i	Conditioning $\kappa_2(C)$	$\sqrt{n} \cdot \gamma(C, f)$	Relative error	
			BP-type	GEPP
1	5e+17	4e+16	1e+00	4e+04
2	5e+17	4e+16	1e+00	1e+04
3	5e+17	4e+16	1e+00	6e+03
4	5e+17	6e+15	1e+00	2e+04
5	5e+17	5e+14	1e+00	3e+04
6	5e+17	3e+13	1e+00	1e+05
7	5e+17	2e+12	1e+00	8e+03
8	5e+17	8e+10	1e+00	2e+04
9	5e+17	3e+09	6e-01	3e+03
10	5e+17	9e+07	7e-01	1e+05
11	5e+17	2e+06	2e-03	3e+08
12	5e+17	4e+04	4e-04	1e+09
13	5e+17	5e+02	2e-05	7e+09
14	5e+17	1e+01	3e-07	9e+12
15	5e+17	5e+00	1e-07	3e+12
16	5e+17	4e+00	4e-08	4e+12

TABLE 13. *Left singular vectors for the right-hand sides.*

$A$ , then  $\gamma(A, f)$  can be much smaller than  $\kappa_2(A)$ , thus providing a better bound for  $\kappa_2(A, f)$  in (9.14). Linear systems for which the *Chan-Foulser number* in (9.15) is much smaller than  $\kappa_2(A)$  are called *effectively well-conditioned*. We shall refer to an algorithm as *effectively (forward) stable*, if it is guaranteed to produce high relative accuracy for effectively well-conditioned systems.

It was shown in [BKO95a] imply that the relative error in the solution computed via the new BP-type Cauchy solver is bounded by a multiple of the Chan-Foulser number :

$$(9.16) \quad \frac{\|a - \hat{a}\|_\infty}{\|a\|_\infty} \leq (19n + 2)\sqrt{n} \cdot \gamma(C(x, y), f) \cdot u + \mathcal{O}(u^2).$$

This means that

*the new BP-type Cauchy solver is effectively stable for totally positive Cauchy systems.*

Table 13 below illustrates these results with a computed example, in which we solved sixteen linear systems with  $16 \times 16$  Hilbert coefficient matrix using its seven left singular vectors  $u_k$  for the right-hand sides.

The numerical data do not demonstrate any dependence of the accuracy of Gaussian elimination with complete pivoting ( GECP ) upon the direction of the right-hand side vector. Moreover, in all 16 cases the GECP algorithm does not produce even one correct digit in the solution from about 7 possible-in-single-precision, consistent with the large condition number of the coefficient matrix, displayed in the second column.

The same table 13 shows that the numerical behavior of the BP-type Cauchy solver indeed depends upon the direction of the right-hand side vector, thus confirming the analytical results in (9.16). Moreover, when the largest eight left singular vectors  $u_1, \dots, u_8$  are used for the right-hand side, then the Chan-Foulser number

$\gamma(C(x, y), f)$ , displayed in the 3-rd column of Table 13, is bigger than the reciprocal of the machine precision,  $\frac{1}{u} \approx 10^8$ . Correspondingly, the BP-type algorithm performs similar to GECP. At the same time, when we used three smallest left singular vectors  $u_{14}$ ,  $u_{15}$  and  $u_{16}$  for the right-hand sides, the Chan-Foulser number  $\sqrt{n} \cdot \gamma(C(x, y), f)$  is of the order of unity and much smaller than  $\kappa_2(C(x, y))$ . Correspondingly, the BP-type Cauchy solver performs much better than GECP, giving now about 7 correct digits from about 7 possible-in-single-precision.

[Here we may note that the solution  $\hat{a}$  of a Vandermonde system computed by the original BP algorithm satisfies

$$(9.17) \quad \frac{\|a - \hat{a}\|_\infty}{\|a\|_\infty} \leq 5n\sqrt{n} \cdot \gamma(V(x), f) \cdot u + \mathcal{O}(u^2),$$

see, e.g., [BKO95a]. The latter bound proves that *the BP algorithm is also effectively stable for totally positive Vandermonde systems*. This conclusion justifies the motivation of [CF88], and gives a theoretical support for the numerical examples in [V93].]

## 10. Pivoting and backward stability of polynomial and rational divided difference schemes. Updating and downdating schemes

**10.1. Relation of the BP algorithm to divided differences.** As was noted, the BP algorithm was based on the decomposition of the inverse of the Vandermonde matrix,

$$(10.1) \quad V(x)^{-1} = U_1 \cdot \dots \cdot U_{n-1} \cdot \tilde{L}_{n-1} \cdot \dots \cdot \tilde{L}_1.$$

The particular form of the upper triangular factors  $U_k$  is not relevant for our purposes here, and we shall be interested only in the structure of the lower triangular factors:

$$(10.2) \quad \tilde{L}_k = \left[ \begin{array}{c|ccccc} I_k & & & & & \\ \hline & 1 & & & & \\ & \overline{x_{k+1}-x_k} & & & & \\ & & \ddots & & & \\ & & & \overline{x_n-x_{n-k}} & & \end{array} \right] \cdot \left[ \begin{array}{c|ccccc} I_{k-1} & & & & & \\ \hline & 1 & & & & \\ & -1 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & & -1 & 1 \end{array} \right].$$

In fact, Björck and Pereyra formulated the first part of their algorithm,

$$(10.3) \quad \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} = \tilde{L}_{n-1} \cdot \dots \cdot \tilde{L}_1 \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix},$$

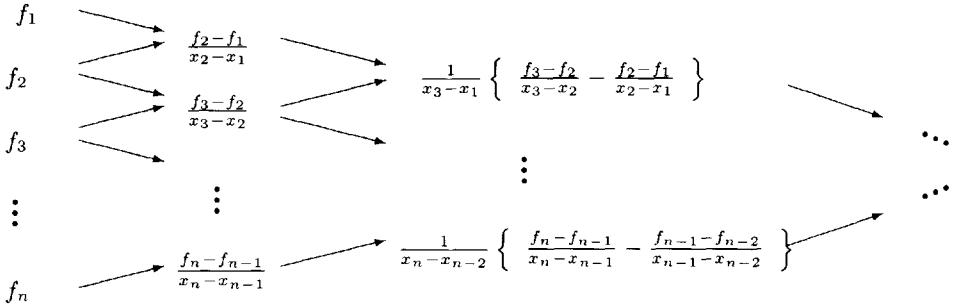
as a matrix interpretation of the well-known divided differences scheme, shown in Figure 2. We call this classical scheme *updating*, to distinguish it from an alternative *downdating* scheme to be introduced later in this section.

The top row of Figure 2 gives us the coefficients  $c_1 = f_1, c_2 = \frac{f_2-f_1}{x_2-x_1}, \dots$ , in the classical Newton representation

$$(10.4) \quad f(x) = \sum_{i=1}^n c_i \cdot \prod_{k=1}^{i-1} (x - x_k)$$

for the interpolating polynomial satisfying

$$f(x_k) = f_k, \quad (k = 1, 2, \dots, n).$$

FIGURE 2. *Updating divided differences.*

**10.2. Downdating divided differences.** As we shall see in a moment, the error analysis for the above classical updating scheme will indicate to certain limitations in using pivoting with this algorithm, so it will be reasonable to look for an alternative. To obtain such an alternative we notice that the formula (10.1) can be obtained by using purely matrix arguments, as just an implementation of Gaussian elimination on  $V(x)$ , cf. with [TG81]. The bidiagonal structure of  $\tilde{L}_k$  means that in this elimination scheme we use the  $k$ -th entry in the first column to eliminate the next  $(k+1)$ -st entry, this variant was called the Neville elimination in [GP92]. An alternative scheme can be obtained by using just standard pivotal scheme of Gaussian elimination on  $V(x)$ , where we use the same (1.1) entry to eliminate all other entries in the first column. As a result we obtain almost the same decomposition

$$(10.5) \quad V(x)^{-1} = U_1 \cdot \dots \cdot U_{n-1} \cdot L_{n-1} \cdot \dots \cdot L_1.$$

but now with slightly different lower triangular factors

$$(10.6) \quad L_k = \begin{bmatrix} 1 & & & \\ & \frac{1}{x_2-x_1} & & \\ & & \ddots & \\ & & & \frac{1}{x_3-x_1} \end{bmatrix} \begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ \vdots & & \ddots & \\ -1 & & & 1 \end{bmatrix}.$$

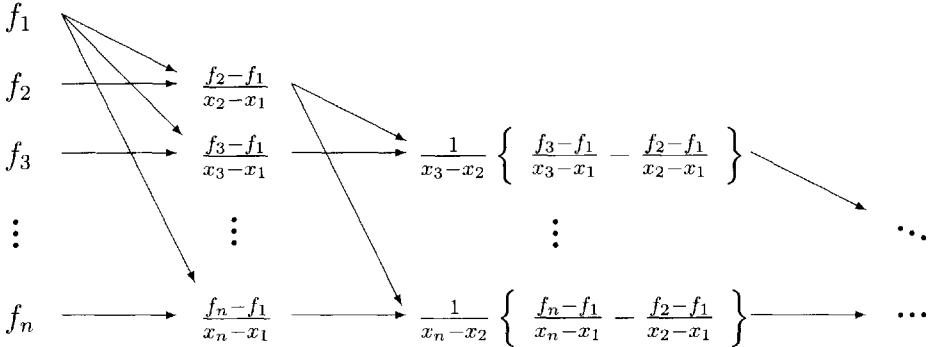
see, e.g., [KO96b]. Since the  $L$  factor the  $LU$  factorization for  $V(x) = LU$  is unique [up to a diagonal factor], the computation

$$\begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} = L_{n-1} \cdot \dots \cdot L_1 \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix},$$

with  $L_k$  as in (10.6) produces the same result, i.e., the vector  $c$  of the coefficients in (10.4), thus giving rise to a new, *downdating* divided difference table:

The justification for the nomenclatures for these two divided differences Tables is given next.

**10.3. Polynomial interpretation.** Denote by  $P_{i_1 i_2 \dots i_k}(x)$  the  $k$ -th degree Newton polynomial interpolating at  $k$  points  $\{x_{i_1}, x_{i_2}, \dots, x_{i_k}\}$ . Then the standard divided differences in Figure 2 have an *updating* nature, recursively computing

FIGURE 3. *Downdating divided differences.*

$P_{i,i+1,\dots,i+k}(x) \rightarrow P_{i,i+1,\dots,i+k+1}(x)$ , see, e.g., [SB80]. For example, the first step updates  $P_i(x) = f_i \rightarrow P_{i,i+1}(x) = f_i + \frac{f_{i+1}-f_i}{x_{i+1}-x_i}(x-x_i)$  for  $i = 1, 2, \dots, n-1$ . Note that the interpolation data remain unchanged.

In contrast, the non-standard divided differences in Figure 3 has a *downdating* nature, recomputing the interpolation data as follows. The desired Newton interpolating polynomial clearly has the form  $f(x) = f_1 + (x-x_1) \cdot g(x)$ , with some  $g(x)$ . The Figure 3 starts with the data  $\{x_i, f_i\}_{i=1}^n$  for  $f(x)$ , and then it recomputes the interpolating data  $\{x_i, \frac{f_i-f_1}{x_i-x_1}\}_{i=2}^n$  for  $g(x)$ , i.e., it removes one interpolation point. Then we proceed with  $g(x)$  recursively.

Both divided differences schemes have the same computational complexity  $O(n^2)$  operations, however the downdating scheme in Figure 3 allows us a suitable reordering of the points  $\{x_k\}$  to achieve a better backward stability of the algorithm, as explained next.

**10.4. Backward stability of downdating divided differences.** In fact, the above interpolation interpretation is simpler than its matrix formulation, however the latter seems to be more useful for the analysis of numerical performance, and its provides an insight on how to numerically stabilize the algorithm. Recall that the standard updating divided differences are computed by  $c = \tilde{L}_{n-1} \cdots \tilde{L}_1 \cdot f$ , with  $\tilde{L}_k$  as in (10.2), and that the rounding error analysis for it can be found, e.g., in [Hig96]. It turns out that in the standard model of floating point arithmetic (2.3) the vector  $\hat{c}$  of computed coefficients of (10.4) satisfies

$$(10.7) \quad |f - L \cdot \hat{c}| \leq ((1-u)^{-3(n-1)} - 1) |\tilde{L}_1^{-1}| \cdot \dots \cdot |\tilde{L}_{n-1}^{-1}| \cdot |\hat{c}|.$$

Denoting by

$$(10.8) \quad L = \tilde{L}_1^{-1} \cdot \dots \cdot \tilde{L}_{n-1}^{-1},$$

a lower triangular factor of the Vandermonde matrix  $V(x) = LU$ , we notice that because of possible cancelation the entries this product can be much smaller then the entries of the product of the moduli of the same matrices, i.e.

$$|L| \leq |\tilde{L}_1^{-1}| \cdot \dots \cdot |\tilde{L}_{n-1}^{-1}|.$$

Now observe that for a given interpolation data  $\{x_i, f_i\}_{i=1}^n$  there are many different Newton polynomials of the form (10.4), depending upon different orderings of interpolation points, and that these different forms may have different numerical

properties. It was noted in [Hig96], p.111 that if interpolation points are monotonically ordered,  $x_1 < x_2 < \dots < x_n$ , then the sign pattern of the factors (10.2) prevents any cancelation, allowing one to replace (10.7) by a better bound

$$(10.9) \quad |f - L \cdot \hat{c}| \leq ((1-u)^{-3(n-1)} - 1) |L| \cdot |\hat{c}|.$$

Denoting by  $\hat{f} = L \cdot \hat{c}$  the vector of the values of the *actually computed* Newton polynomial

$$\hat{f}(x) = \sum_{i=1}^n \hat{c}_i \cdot \prod_{k=1}^{i-1} (x - x_k)$$

at interpolation points  $\{x_k\}$ , we can rewrite (10.9) in a more clear form

$$(10.10) \quad |f - \hat{f}| \leq 3nu |L| \cdot |\hat{c}| + O(u^2).$$

Note, however, that  $|L|$  still can be large with monotonic ordering, and that the bidiagonal structure of  $\tilde{L}_k$  do not allow one to formulate the attractive bound (10.10) for an arbitrary ordering.

Now turn to the new downdating divided difference table, for which we can derive a similar to (10.7) bound

$$(10.11) \quad |f - L \cdot \hat{c}| \leq ((1-u)^{-3(n-1)} - 1) |L_1^{-1}| \cdot \dots \cdot |L_{n-1}^{-1}| \cdot |\hat{c}|,$$

but now with the factors of the form (10.6). In contrast to the factors  $\tilde{L}_k$ , whose bidiagonal structure invalidates, in general, the nice bound (10.10), the particular sparsity pattern of  $L_k$  *always* implies (10.9) [because  $|L| = |L_1^{-1}| \cdot \dots \cdot |L_{n-1}^{-1}|$ ]. This means that with the downdating divided differences table we have a complete freedom of rearrangements of the  $\{x_i\}$ , and can use this freedom to reduce the size of  $|L|$  in (10.10) to achieve a better numerical stability. Recalling that

$$L = \check{L} \cdot \text{diag}(1, (x_2 - x_1), \dots, \prod_{k=1}^{n-1} (x_n - x_k)).$$

where  $\check{L}$  is the *unit* [i.e., with ones on the main diagonal] lower triangular factor in the LU decomposition  $V_P(x) = \check{L}U$ , and taking an advantage of the observation that rearrangement of  $\{x_i\}$  is equivalent to a row permutation of  $V_P(x)$ , we arrive at the conclusion that the *partial pivoting* ordering guarantees that the entries of  $\check{L}$  is less than unity in magnitude. Therefore (10.11) yields a better backward stability of the downdating divided differences :

$$(10.12) \quad |f_k - \hat{f}_k| \leq 3nu \sum_{i=1}^k |\hat{c}_i| \cdot |\prod_{j < i} (x_i - x_j)| + O(u^2).$$

A comparison of the bounds (10.10) and (10.12) suggests that the downdating divided differences table with partial pivoting is more attractive then the [classical] updating one, as long as the backward stability is concerned. Moreover, partial pivoting for Vandermonde matrices is equivalent to Leja ordering (1.2) for which we have a fast  $O(n^2)$  algorithm [Hig90], so its incorporation will not essentially slow down the overall algorithm.

**10.5. Rational divided differences.** The above discussion concerns two different algorithms to find the interpolating polynomial in the Newton form, i.e., in this problem we were looking for an interpolant the linear space with the basis

$$(10.13) \quad \{1, (x - x_1), (x - x_1)(x - x_2), \dots, (x - x_1) \cdots \cdots (x - x_{n-1})\}.$$

Other bases of functions, not necessarily polynomials, are also of interest. Say, finding an interpolant with respect to the basis

$$(10.14) \quad \left\{ \frac{1}{x - y_1}, \dots, \frac{1}{x - y_n} \right\}$$

[i.e., among rational functions with the fixed poles  $\{y_k\}$ ] is equivalent to solving a Cauchy linear system of equations. Other bases in this space of rational functions can be better conditioned than (10.14), motivating us to look more closely at a rational analog of the Newton basis, i.e., at the following combination of (10.13) and (10.14):

$$\left\{ \frac{1}{x - y_1}, \frac{x - x_1}{(x - y_1)(x - y_2)}, \dots, \frac{(x - x_1) \cdots \cdots (x - x_{n-1})}{(x - y_1) \cdots \cdots (x - y_{n-1})(x - y_n)} \right\}.$$

It turns out that the results on Cauchy matrices in Sec. 8 and 9 readily suggest two immediate algorithms to compute a rational interpolant in the Newton-like form:

$$(10.15) \quad f(x) = \sum_{k=1}^n c_k \cdot \frac{(x - x_1) \cdots \cdots (x - x_{k-1})}{(x - y_1) \cdots \cdots (x - y_{k-1})} \frac{1}{(x - y_k)},$$

Indeed, it can be shown that the situation is totally analogous to the relation of (10.3) to polynomial divided differences, so computing the coefficients of a rational function of the form (10.15) satisfying  $f(x_k) = f_k$  is equivalent to solving a linear system of equations

$$\begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} = L^{-1} \cdot \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix},$$

where  $L$  is a lower triangular factor of  $C(x, y) = LU$ . Now just recall that in Lemmas 9.1 and 8.2 we specified two different decompositions for the desired lower triangular factor, both of the form

$$(10.16) \quad L^{-1} = D \cdot L_{n-1} \cdots \cdots L_1.$$

As the reader already guessed, these two decompositions give rise to two schemes, *updating* and *downdating*, to compute the vector  $\{c_k\}$  of *rational divided differences*. Specifically, the BP-type decomposition (9.5) [proven via the Neville elimination scheme] suggests a bidiagonal form for the  $L_k$ 's in (10.16):

Factors for the updating rational divided differences

$$(10.17) \quad L_k = \left[ \begin{array}{c|ccccc} I_k & & & & & \\ \hline & \frac{1}{x_{k+1}-x_1} & & & & \\ & & \frac{1}{x_{k+2}-x_2} & & & \\ & & & \ddots & & \\ & & & & \frac{1}{x_n-x_{n-k}} & \end{array} \right] \times$$

$$\times \left[ \begin{array}{c|ccccc} I_{k-1} & & & & & \\ \hline & 1 & & 0 & & \\ & -(x_1 - y_k) & (x_{k+1} - y_k) & & & \\ & & \ddots & \ddots & & \\ & & & & -(x_{n-k} - y_k) & (x_n - y_k) \end{array} \right].$$

Similarly, the decomposition (8.4) [obtained via the usual pivotal scheme of Gaussian elimination] suggests slightly different  $L_k$ 's for (10.16):

Factors for the downdating rational divided differences

$$(10.18) \quad L_k = \left[ \begin{array}{c|ccccc} I_{k-1} & & & & & \\ \hline & 1 & & & & \\ & & \frac{1}{x_{k+1} - x_k} & & & \\ & & & \ddots & & \\ & & & & \frac{1}{x_n - x_k} & \end{array} \right] \left[ \begin{array}{c|ccccc} I_{k-1} & & & & & \\ \hline & 1 & & & & \\ & -1 & 1 & & & \\ & & \vdots & \ddots & & \\ & & -1 & & & 1 \end{array} \right] \left[ \begin{array}{c|ccccc} I_{k-1} & & & & & \\ \hline & (x_k - y_k) & & & & \\ & & \ddots & & & \\ & & & & & (x_n - x_k) \end{array} \right]$$

The computational complexity of the updating and downdating rational divided difference schemes is the same, but we next show that a fast implementation of partial pivoting [e.g., as a predictive partial pivoting of Sec. 8.5] again yields much smaller backward error bounds for the downdating scheme.

**10.6. Backward stability of rational downdating divided differences.**

Essentially the same backward error bound (10.7) of polynomial divided differences can be obtained for their rational counterparts:

$$(10.19) \quad |f - L \cdot \hat{c}| \leq d_n u |L_1^{-1}| \cdot \dots \cdot |L_{n-1}| \cdot |D^{-1}| \cdot |\hat{c}|,$$

where  $\hat{c}$  denotes the vector of *computed* rational divided differences.  $L$  is a lower triangular factor of the Cauchy matrix  $C(x, y) = LU$ :

$$L^{-1} = D \cdot L_{n-1} \cdot \dots \cdot L_1.$$

and  $d_n$  is a polynomial of small degree in  $n$  [whose exact form is not relevant for our analysis]. The error bound (10.19) is valid for the both updating and downdating rational divided differences, and it is quite pessimistic, because of the absolute values on the right-hand-side. Denoting by  $\hat{f} = L \cdot \hat{c}$  the vector of the values at  $\{x_k\}$  of the *actually computed* rational interpolant

$$f(x) = \sum_{k=1}^n \hat{c}_k \cdot \frac{(x - x_1) \cdot \dots \cdot (x - x_{k-1})}{(x - y_1) \cdot \dots \cdot (x - y_{k-1})} \frac{1}{(x - y_k)}$$

at  $\{x_k\}$ , we can use the same arguments as in Sec. 10.4 to motivate that the bound of the form

$$(10.20) \quad |f - \hat{f}| \leq d_n u |L| \cdot |\hat{c}|.$$

would be much better than (10.19). In fact, the other arguments of Sec 9.4 carry over to the rational divided differences as well: for the updating scheme we are able

to remove moduli in (10.19) and to deduce (10.20), but we have to assume

$$y_k < x_1 < x_2 < \dots < x_n.$$

So, even if  $y_k$ 's are all less than  $x_k$ 's, the desired bound (10.20) was established for the updating scheme with only one particular ordering, i.e., only for one Newton-like representation of the rational interpolant  $f(x)$ . But unfortunately this particular ordering unfortunately does not guarantee a small  $|L|$ .

In contrast, the situation with downdating rational divided differences is favorable, and for any ordering we may wish we shall have the identity

$$|L| = |L_1^{-1}| \cdot \dots \cdot |L_{n-1}^{-1}| \cdot |D^{-1}|,$$

always implying the desired bound (10.20). Of course, the first candidate to look at is the partial pivoting ordering, because it produces the unit lower triangular factor  $\check{L}$  [in the LU factorization of  $C(x, y)$ ] with the entries less than unity. Our  $L$  and the unit  $\check{L}$  are related as

$$\begin{aligned} L = \check{L} \cdot \text{diag} \{ & \frac{1}{x_1 - y_1}, \frac{x_2 - x_1}{(x_2 - y_1)(x_2 - y_2)}, \dots \\ & \dots, \frac{(x_n - x_1) \cdot \dots \cdot (x_n - x_{n-1})}{(x_n - y_1) \cdot \dots \cdot (x_n - y_{n-1})(x_n - y_n)} \}, \end{aligned}$$

so that (10.20) implies a better backward stability:

$$|f_k - \hat{f}_k| \leq d_n u \cdot \sum_{k=1}^n |\hat{c}_k| \cdot \frac{|(x_k - x_1) \cdot \dots \cdot (x_k - x_{k-1})|}{|(x_k - y_1) \cdot \dots \cdot (x_k - y_{k-1})|} \frac{1}{|(x_k - y_k)|} + O(u^2).$$

for the downdating rational divided differences. To sum up, for the two rational divided difference schemes the conclusion is exactly the same as for the polynomial ones: the downdating scheme with partial pivoting can be preferable if the backward stability is concerned. Finally recall that we have a fast  $O(n^2)$  predictive partial pivoting algorithm of Sec. 8.5 to use with the downdating rational divided difference scheme without slowing it down.

## 11. Concluding remarks

Analysis of the numerical behavior of array algorithms is one of the central themes in numerical linear algebra, perhaps the most important for the design of quality software for solving a wide variety of the associated applied problems. After about two decades of close attention, we now have a good understanding of the numerical properties of many standard algorithms for basic linear algebra problems, and we now have many stabilizing techniques, i.e., recipes to improve the accuracy. The most classical and important among such problems is, perhaps, the analysis of the performance of the Gaussian elimination procedure in the finite-precision-arithmetic. One of the early important results in this direction was the Wilkinson's error analysis leading to a recipe to use pivoting to numerically stabilize this classical algorithm.

In a different direction, reformulating a certain applied problem as a linear algebra problem, we quite often have to form a square array which is much larger than the actual amount of the input parameters. In other words, matrices appearing in applications are, as a matter of fact, structured, in the sense that they are defined by a small number of parameters. Different applications give rise to many different patterns of structure, including Toeplitz, Hankel, Vandermonde, Cauchy

matrices, their confluent versions, Bezoutians, controllability, Schur-Cohn, Rouz-Hurwitz matrices, and many other patterns of structure. Exploiting the structure of a matrix to speedup the computation is another well-known topic in numerical linear algebra, but for the time being the analysis of accuracy of such structure-exploiting fast algorithms seems to be still behind the results obtained for their general purpose counterparts. As a result, much less stabilizing techniques have been proposed, and often a user has no other choice but to use slow structure-ignoring, but accurate, algorithms. Briefly, to sacrifice speed to achieve accuracy.

In this paper we reviewed recent progress made at the intersection of the above two areas. The point is that many well-known and new algorithms for important classes of applied problems admit matrix interpretations as a Gaussian elimination procedure, but efficiently implemented for a matrix with a certain pattern of structure. This association suggests to use pivoting, so successful for the usual Gaussian elimination procedure, to improve the accuracy of its fast structure-exploiting relatives. We described recently developed fast algorithms for factorization of Cauchy and Cauchy-like matrices, appearing in various tangential interpolation problems [we used an example of suboptimal rational matrix Nehari problem to show how to solve interpolation problems belonging to this class]; presented realistic algorithms to factorize general and  $J$ -unitary rational matrix functions, obtained various divided difference schemes [including polynomial and rational, updating and downdating]; developed Toeplitz-like and Toeplitz-like solvers through the transformation-to-Cauchy approach. In all these cases we suggest structure-exploiting stabilizing techniques, whose matrix interpretation is pivoting, efficiently implemented for a particular pattern of structure. We found, either numerically or through rounding error analysis, that such fast pivoting is crucial to computing not only fast, but also accurate, solutions. These results will contribute to the design of a comprehensive quality software package for solving structured matrix problems and applications.

## 12. Appendix. Some results not related to pivoting

The primary focus of this survey is various pivoting techniques for fast algorithms. The algorithms themselves were briefly described in the main text above only when they were needed. In this Appendix we describe several recent generalizations and improvements of some of those algorithms: the discussion here is not immediately related to pivoting. We consider here four such extensions in Sec. 12.1-12.4. Throughout this Appendix we consider strongly regular matrices, i.e., whose all leading minors are nonzero.

**12.1. The Hessenberg Displacement Structure.** Several basic classes of structured matrices listed in Table 3 were complemented in Sec 7.7.1 with another class of polynomial Hankel-like matrices  $R$  studied earlier in [KO96] and defined in Sec 7.7.1 via a displacement equation involving Hessenberg matrices. Before proceeding further, let us first consider two simple examples.

*Example 12.1.1.* Toeplitz-plus-Hankel-like matrices. Let us consider the Hessenberg matrix

$$Z + Z^T = \begin{bmatrix} 0 & 1 & 0 & \cdots & \cdots & 0 \\ 1 & 0 & 1 & & & \\ 0 & 1 & \ddots & \ddots & \ddots & \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ & & & \ddots & 0 & 1 \\ \vdots & & & & 0 & 1 \\ 0 & \cdots & \cdots & 0 & 1 & 0 \end{bmatrix},$$

where  $Z$  denotes, as above, the lower shift matrix. It is easy to see that the corresponding displacement rank of any Toeplitz-plus-Hankel matrix  $T + H$  does not exceed 4:

$$(Z + Z^T)(T + H) - (T + H)(Z + Z^T) =$$

$$\left[ \begin{array}{cc} \square & \square \\ \square & \square \\ \downarrow & \square \\ \square & \square \\ \uparrow & \square \\ \square \end{array} \right] - \left[ \begin{array}{ccccc} \square & & & & \\ \square & \Rightarrow & \square & \Leftarrow & \square \\ \square & & & & \square \end{array} \right] = \left[ \begin{array}{c|ccccc} * & * & \cdots & * & * \\ \hline * & & & & & * \\ * & & & & & * \\ \vdots & & & 0 & & \vdots \\ * & & & & & * \\ * & * & \cdots & * & * & * \end{array} \right]$$

*Example 12.1.2.* Generalized Bezoutians. Let us consider another Hessenberg matrix

$$(12.1) \quad H_Q = \begin{bmatrix} a_{01} & a_{02} & \cdots & \cdots & a_{0,n} \\ a_{11} & a_{12} & \cdots & \cdots & a_{1,n} \\ 0 & a_{22} & \cdots & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & a_{n-1,n-1} & a_{n-1,n} \end{bmatrix}$$

and the associated system  $\{Q_k\}$  of polynomials<sup>8</sup> satisfying

$$(12.2) \quad x \cdot Q_{k-1}(x) = a_{k,k} \cdot Q_k(x) + \dots + a_{0,k} \cdot Q_0(x).$$

Following [GO94a] let us consider the generalized Bezoutians  $B_Q = [b_{ij}]$  whose entries are obtained from  $\frac{a(x)b(y)-a(y)b(x)}{x-y} = \sum b_{ij}Q_i(x)Q_j(y)$ . It can be shown that the corresponding displacement rank of  $B_Q$  is at most two:

$$(12.3) \quad \text{rank}(H_Q^T B_Q - B_Q H_Q) \leq 2$$

*Triangularity condition.* The above two examples suggest to develop algorithms for computing the triangular factorization  $R = LU$  for what we suggested to call polynomial Hankel-like matrices satisfying

$$FR - RA = GB, \quad G \in \mathbb{C}^{n \times \alpha}, B \in \mathbb{C}^{\alpha \times n}$$

with an upper Hessenberg  $F$  and a lower Hessenberg  $A$ . Section 7.7 specified a way of transforming such a polynomial Hankel-like matrix  $R$  into a Cauchy-like matrix, but not a direct algorithms for computing  $R = LU$ . Moreover, a recent paper [K99] claims that this is not possible: “*The restriction that  $\{F, A^T\}$  are lower triangular is the most general one that allows recursive triangular factorization (cf. a result in [LAK86]).*” Recall that theorems 4.1, 4.2 relate the recursive factorization of  $R$  to the factorization of  $W_1(z)$ , and one of the conditions of these theorems is that the matrices  $\{A, A^\times\}$  or matrices  $\{A_\pi, A_\zeta\}$  are triangular. Hence the claim of

<sup>8</sup>As was mentioned in Sec. 7.7.1  $H_Q$  was called *the confederate matrix* of  $\{Q_k\}$  in [MB79].

[K99] seems to match the last sentence in theorem 4.1: “*Each minimal factorization of  $W_1(z)$  can be obtained in this way.*” The latter statement cannot be refuted; however, the claim of [K99] is formally incorrect. The words in [K99] seem to be poorly chosen, since a fast recursive algorithm for  $R = LU$  can be obtained in a different way, not necessarily as a matrix interpretation of the above theorems. Moreover, a paper [SLAK93] contains such a recursive factorization algorithm for a sum of quasi-Toeplitz and a quasi-Hankel matrices. A much more general result is stated next.

*Recursive triangular factorization.* The following counterpart of lemma 3.1 is the basis for the desired algorithm.

**THEOREM 12.1.** [**HO01**] *Let  $R^{(1)}$  satisfy*

$$F^{(1)}R^{(1)} - R^{(1)}A^{(1)} = G^{(1)}(B^{(1)})^T.$$

*and let the latter matrices be partitioned as*

$$\begin{aligned} R^{(1)} &= \begin{bmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{bmatrix}, F^{(1)} = \begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix}, A^{(1)} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \\ G^{(1)} &= \begin{bmatrix} G_1 \\ G_2 \end{bmatrix}, B^{(1)} = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}. \end{aligned}$$

*If  $R_{11}$  is nonsingular then the Schur complement  $R^{(2)} = R_{22} - R_{21}R_{11}^{-1}R_{12}$  satisfies*

$$F^{(2)}R^{(2)} - R^{(2)}A^{(2)} = G^{(2)}(B^{(2)})^T,$$

$$\text{with } \begin{bmatrix} G^{(2)} = G_2 - QG_1 & (B^{(2)})^T = B_2^T - B_1^TP \\ A^{(2)} = A_{22} - A_{21}P & F^{(2)} = F_{22} - QF_{12} \end{bmatrix}$$

*where we denote*

$$P := R_{11}^{-1}R_{12}, \quad Q := R_{21}R_{11}^{-1}.$$

One can use this lemma to design a fast algorithm in which manipulating on  $R^{(1)}$  is replaced by updating the six matrices  $\{P, Q, F^{(1)}, A^{(1)}, G^{(1)}, B^{(1)}\}$  (we can now call them a generator of  $R_1$ ). Similarly to (3.4) such a fast algorithm can be visualized as follows.

$$\begin{array}{ccc} R^{(1)} & \longrightarrow & R^{(2)} \\ \downarrow & & \downarrow \\ \{l_1, u_1, F^{(1)}, A^{(1)}, G^{(1)}, B^{(1)}\} & \longrightarrow & \{l_2, u_2, F^{(2)}, A^{(2)}, G^{(2)}, B^{(2)}\} \end{array} \longrightarrow \dots$$

where  $l_1$  and  $u_1$  are the first column and the top row of  $R_1$ . In general, the latter algorithm is fast only for the low complexity matrices  $\{F^{(1)}, A^{(1)}\}$  such as companion, tridiagonal, comrade or unitary Hessenberg. See [HO01] for more details.

*A polynomial interpretation: the generalized Euclid Algorithm.* To match the matrix in the Example 12.1.2 and theorem 12.1 Let us rewrite (12.3) as

$$F^{(1)}R^{(1)} - R^{(1)}A^{(1)} := (JH_Q^T J)(JB_Q J) - (JB_Q J)(JH_Q J),$$

where  $J$  is a counter-identity matrix [having 1’s on the main counter-diagonal].

Let look closer at updating the Hessenberg matrix

$$A^{(1)} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \longrightarrow A^{(2)}$$

which can be graphically represented as follows:

(12.4)

$$A^{(2)} = A_{22} - A_{21}P = \begin{bmatrix} \times & \times & \cdots & \cdots & \times \\ \times & \times & \cdots & \cdots & \times \\ 0 & \times & \times & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \times \\ 0 & \cdots & 0 & \times & \times \end{bmatrix} - \begin{bmatrix} \times \\ 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times & \times \end{bmatrix}$$

To sum up, the latter update modifies only the top row of  $A^{(1)}$ . It is easy to see from (12.1) that the top row of  $A^{(1)} = JH_QJ$  corresponds to only the last polynomial  $Q_k(x)$  [to see this, just have a closer look at the coefficients in (12.2)]. Hence the (12.4) on the swapped generalized Bezoutian  $JB_QJ$  is a form of a polynomial recursion, and it can be shown [OO02] that it corresponds to the Euclid algorithm for polynomials represented in the basis  $\{Q_k\}$ .

**12.2. Backward stable factorization of Hankel-like matrices.** Let us consider a positive definite Hankel matrix defined in Table 3. The structure-ignoring slow  $O(n^3)$  Cholesky algorithm computes  $H \approx \widehat{L}\widehat{L}^T$  with a small backward error:

$$(12.5) \quad \|H - \widehat{L}\widehat{L}^T\|_2 < O(u)\|H\|_2, \quad \text{where } u \text{ is the machine precision.}$$

There are many fast algorithms for  $H$ , but none of them is known to yield (12.5), and for most of them it is known that it is hopeless. We describe next the first fast algorithm which is provable as backward stable as Cholesky in (12.5).

The algorithm is based on the observation [cf. with Table 4] that the displacement rank of  $H$  does not exceed two:

$$(12.6) \quad ZH - HZ^T = \left[ \begin{array}{c|ccc} 0 & -h_0 & -h_1 & -h_2 \\ \hline h_0 & & & \\ h_1 & & & \\ h_2 & & & \end{array} \right] =$$

$$\left[ \begin{array}{cc} h_0 & 0 \\ 0 & h_0 \\ 0 & h_1 \\ 0 & h_2 \end{array} \right] \left[ \begin{array}{cc} 0 & -1 \\ 1 & 0 \end{array} \right] \left[ \begin{array}{cc} h_0 & 0 \\ 0 & h_0 \\ 0 & h_1 \\ 0 & h_2 \end{array} \right]^T = GJG^T$$

Since we consider positive definite  $H$ , no pivoting is necessary, and a fast algorithms can be designed directly using the recursions (3.8) of Lemma 3.1. However, numerical examples in [OS99, OS01] show that such a straightforward approach can yield a lack of backward accuracy. Such a poor numerical performance is not surprising given the fact that positive definite Hankel matrices are exponentially ill-conditioned [T94], and hence are quite delicate objects. The error analysis shows that the algorithm can be stabilized by preceding each recursion step (3.8) by a modification of the generator matrix

$$G \leftarrow G\Theta, \quad \text{where } \Theta J\Theta^T = J, \quad \text{so that } G\Theta J\Theta^T G^T = GJG^T \quad \text{in (12.6).}$$

One “good”  $\Theta$  has the form  $\Theta = \Theta_1\Theta_2$ . Here  $\Theta_1 = \begin{bmatrix} d & 0 \\ 0 & \frac{1}{d} \end{bmatrix}$  produces the new  $G$  whose two columns have the same norm. The second  $\Theta_2 = \begin{bmatrix} c & -s \\ s & c \end{bmatrix}$  is an

orthogonal matrix producing a new  $G$  whose  $(1,2)$  entry is zero [the so-called proper form]. The quite involved error analysis guarantees that such a seemingly trivial modification results in the desired favorable backward accuracy (12.5). Thus, one should not sacrifice the speed to have accuracy. We refer to [**OS99**, **OS01**] for a little bit more involved analysis of the algorithm for positive definite Hankel-like matrices yielding the same high backward accuracy (12.5), as well as for a new look-ahead technique.

**12.3. Superfast algorithms for structured matrices.** The algorithms described above are fast, requiring  $O(n^2)$  arithmetic operations. In this section we describe superfast algorithms that require much less: from  $O(n \log^2 n)$  to  $O(n \log^2 n)$  depending on a particular pattern of structure. Such algorithms are based on the now standard divide-and-conquer technique that can be graphically described by the figure 4.

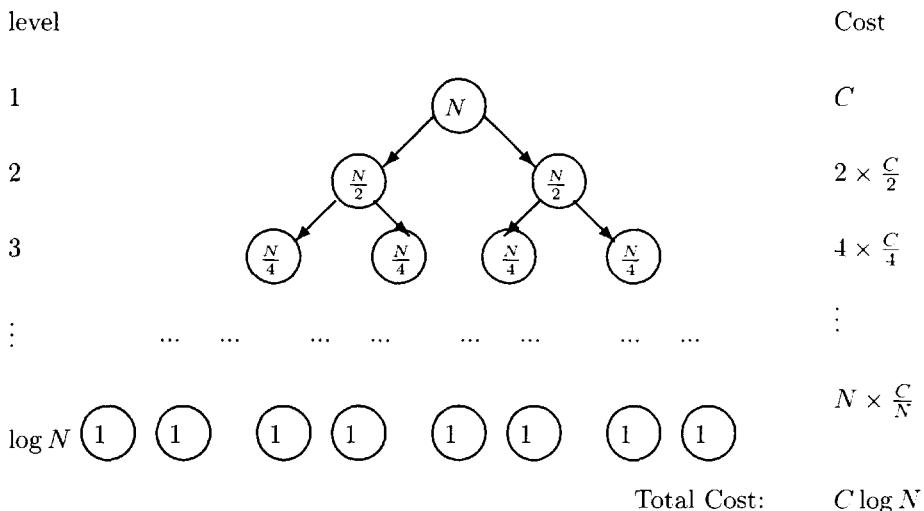


FIGURE 4. Divide-and-conquer approach.

The latter shows that if reducing a problem of the size  $N$  to the two similar problems of the size  $N/2$  requires  $C$  operations then one can proceed recursively which results in a superfast algorithm with the overall cost of only  $O(C \log N)$  operations. Perhaps the first application of this idea to structured matrices was made in [M80] and [BA80] where the first superfast algorithm for Toeplitz-like matrices was designed. The next lines simplify their method and extend it to Hankel-like, Vandermonde-like and Cauchy-like matrices.

The reader may now have a new look at lemma 3.1 to see that it can be applied to the recursive block Schur complementation

$$R_1 = \begin{bmatrix} R_{11} & R_{21} \\ R_{21} & R_{22} \end{bmatrix} = \begin{bmatrix} I & 0 \\ R_{21}R_{11}^{-1} & I \end{bmatrix} \begin{bmatrix} R_{11} & 0 \\ 0 & R_2 \end{bmatrix} \begin{bmatrix} I & R_{11}^{-1}R_{21} \\ 0 & I \end{bmatrix}$$

A matrix	The cost of matrix-vector product
Toeplitz-like and Hankel-like	$O(n \log n)$
Vandermonde-like matrices	$O(n \log^2 n)$
Cauchy-like matrices	$O(n \log^2 n)$

TABLE 14. *Costs of computing the matrix-vector product. See [GO94b] for the description of the algorithms.*

A matrix	The cost of recursive triangular factorization
Toeplitz-like and Hankel-like	$O(n \log^2 n)$
Vandermonde-like matrices	$O(n \log^3 n)$
Cauchy-like matrices	$O(n \log^3 n)$

TABLE 15. *Costs of computing the matrix-vector product. See [OP98] for the description of the algorithms.*

A problem	The cost
Computing matrix-vector product	$O(n \log n \cdot [1 + \sum_{k=1}^r \frac{m_k}{n} \log \frac{n}{m_k} + \sum_{k=1}^s \frac{t_k}{n} \log \frac{n}{t_k}])$
Computing recursive triangular factorization	$O(n \log^2 n \cdot [1 + \sum_{k=1}^r \frac{m_k}{n} \log \frac{n}{m_k} + \sum_{k=1}^s \frac{t_k}{n} \log \frac{n}{t_k}])$

TABLE 16. *Costs for confluent Cauchy-like matrices. See [OS99] for the description of the algorithms. Here  $\{m_k\}$  are the sizes of the Jordan blocks of  $A_\zeta$  and  $\{t_k\}$  are the sizes of the Jordan blocks of  $A_\pi$ , , see, e.g., (12.8).*

where the new generator of  $R_2$  needs to be computed via

(12.7)

$$\begin{bmatrix} 0 \\ G_2 \end{bmatrix} = G_1 - \begin{bmatrix} I_m \\ R_{21}R_{11}^{-1} \end{bmatrix} \cdot G_{11}, \quad \begin{bmatrix} 0 & B_2 \end{bmatrix} = B_1 - B_{11} \cdot \begin{bmatrix} I_m & R_{11}^{-1}R_{21} \end{bmatrix},$$

The latter formulas represent exactly one divide-and-conquer step, so their cost  $C$  determines the overall cost  $O(C \log n)$  of the algorithm. Let us therefore look more closely at (12.7). They require computing matrix-vector products for the matrices of the same type as  $R_1$  [e.g., its submatrices, their inverses, Schur complements, etc.]. We therefore list such costs in the Table 14.

Table 14 yields the results in Table 15 that lists the overall costs of the divide-and conquer algorithms for recursive triangular factorization.

*General case. Confluent Cauchy-like matrices.* The above results concern Hankel-like, Vandermonde-like and Cauchy-like matrices. Looking at their definitions in Table 4 we see that for them the auxiliary matrices  $\{F, A\}$  can be chosen to be either shift  $Z$  or a diagonal  $D_x$  [either just one Jordan block or  $n$  Jordan

Toeplitz matrices	convolution	$O(n \log n)$
Hankel matrices	convolution	$O(n \log n)$
Vandermonde matrices	multipoint polynomial evaluation	$O(n \log^2 n)$
DFT matrices (i.e., Vandermonde matrices with special nodes)	discrete Fourier transform	$O(n \log n)$
inverse Vandermonde matrices	polynomial interpolation	$O(n \log^2 n)$
Cauchy matrices	multipoint rational evaluation	$O(n \log^2 n)$
inverse Cauchy matrices	rational interpolation	$O(n \log^2 n)$

TABLE 17. *Connection between fundamental algorithms and structured matrix-vector multiplication*

blocks]. This motivates one to consider a general case when  $R$  satisfies

$$(12.8) \quad \text{rank}(A_\zeta^T R - RA_\pi) \ll n$$

where  $\{A_\zeta, A_\pi\}$  are matrices in the Jordan canonical form. Such matrices were called *confluent Cauchy-like* matrices in [OS99, OS01]. In latter papers one can find the algorithms whose costs are listed in Table 16. Notice that the cost in Table 16 satisfies:

$$O(n \log^2 n) \leq O(n \log^2 n \cdot [1 + \sum_{k=1}^r \frac{m_k}{n} \log \frac{n}{m_k} + \sum_{k=1}^s \frac{t_k}{n} \log \frac{n}{t_k}]) \leq O(n \log^3 n)$$

The cost on the left-hand side occurs for the case of one Jordan block  $n = m_1$ , and the bound on the right-hand side corresponds to the case of  $n$  simple Jordan blocks  $m_k = 1$ .

We conclude this subsection with mentioning that fast matrix-vector multiplication algorithms for Hankel, Vandermonde, Cauchy matrices and their inverses admit well-known polynomial interpretations listed in Since the class of confluent Cauchy-like matrices is fairly general, including Hankel, Vandermonde, Cauchy matrices and their inverses hence the algorithm of [OS99, OS01] can be considered as a generalization of all these algorithms in Table 17.

**12.4. Superfast confluent rational tangential interpolation.** The results of the previous subsection yield superfast interpolation algorithms described next. Let us consider the following generalization of the simplest homogeneous tangential interpolation problem considered in Sec 4.2.

Tangential confluent rational interpolation problem.

$r$  distinct points  $\{z_k\}$  in the open right-half-plane  $\Pi^+$ ,  
with their multiplicities  $\{m_k\}$ .

Given:  $r$  nonzero chains of  $N \times 1$  vectors  $\{x_{k,1}, \dots, x_{k,m_k}\}$ ,  
 $r$  nonzero chains of  $M \times 1$  vectors  $\{y_{k,1}, \dots, y_{k,m_k}\}$ .

Construct: a rational  $N \times M$  matrix function  $F(x)$  such that

- (1)  $F(z)$  is *analytic* inside the right half plane (i.e.. all the poles are in the left half plane).

(2)  $F(z)$  is *passive*, which by definition means that

$$(12.9) \quad \sup_{z \in \Pi^+ \cup i\mathbf{R}} \|F(z)\| \leq 1.$$

(3)  $F(z)$  meets the tangential *confluent* interpolation conditions ( $k = 1, 2, \dots, r$ ):

$$\begin{bmatrix} x_{k1} & \dots & x_{k,m_k} \end{bmatrix} \begin{bmatrix} F(z_k) & F'(z_k) & \dots & \frac{F^{(m_k-1)}(z_k)}{(m_k-1)!} \\ 0 & F(z_k) & \ddots & \vdots \\ \vdots & & \ddots & F'(z_k) \\ 0 & \dots & \dots & F(z_k) \end{bmatrix} =$$

$$(12.10) \quad \begin{bmatrix} y_{k,1} & \dots & y_{k,m_k} \end{bmatrix}.$$

Clearly, the simplest homogeneous tangential interpolation problem considered in Sec 4.2 is a special case. Here are two more.

*Example 12.4.1. The tangential Nevanlinna-Pick problem.* In the case of simple multiplicities  $m_k = 1$  the interpolation condition (12.10) reduces to the usual tangential (i.e.,  $x$ 's and  $y$ 's are vectors) interpolation condition

$$x_k \cdot F(x_k) = y_k$$

which in the case of the scalar  $F(z)$  further reduces to the familiar interpolation condition of the form

$$F(x_k) = \frac{y_k}{x_k}.$$

*Example 12.4.2. The Caratheodory-Fejer problem.* Let the vectors  $\{x_{k,j}\}$  are just the following scalars:

$$\begin{bmatrix} x_{k,1} & \dots & x_{k,m_k} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix}.$$

Clearly, in this case (12.10) this is just a Hermite-type rational passive interpolation problem

$$F(z_k) = y_{k,1}, \quad F'(z_k) = y_{k,2}, \quad \dots \quad \frac{F^{(m_k-1)}(z_k)}{(m_k-1)!} = y_{k,m_k}.$$

The monograph [BGR90] contains the solution to the tangential confluent interpolation problem of the form

$$\Theta(z) = I + C_\pi(zI - A_\pi)^{-1} R^{-1} B_\zeta.$$

with a confluent Cauchy-like matrix  $R$ . A combination of their result, the results in Sec. 4.4 and 4.5 as well as the results of Sec. 12.2 allowed us to design a superfast algorithm for recursive factorization of rational matrix functions  $\Theta(z) = W_1(z)W_2(z) = (W_{11}(z)W_{12}(z)) \cdot (W_{21}(z)W_{22}(z)) = \dots$ . The analogue of (4.19) can be visualized as in Figure 5. Table 18 lists the costs for the three above interpolation problems.

We finally note that the cost of the confluent tangential interpolation problem is bounded as

$$O(n \log^2 n) \leq O(n \log^2 n \cdot [1 + \sum_{k=1}^r \frac{m_k}{n} \log \frac{n}{m_k} + \sum_{k=1}^s \frac{t_k}{n} \log \frac{n}{t_k}]) \leq O(n \log^3 n),$$

The problem	The cost
Tangential confluent	$O(M(n) \log n \cdot [1 + \sum_{k=1}^r \frac{m_k}{n} \log \frac{n}{m_k}])$
Nevanlinna-Pick	$O(n \log^3 n)$
Caratheodori-Fejer	$O(n \log^2 n)$

TABLE 18. *Costs of solving tangential interpolation problems. See [OS99, OS01] for the description of the algorithms.*

where the cost on the left-hand side occurs for the Caratheodory-Fejer case, and the bound on the right-hand side corresponds to the Nevanlinna-Pick case.

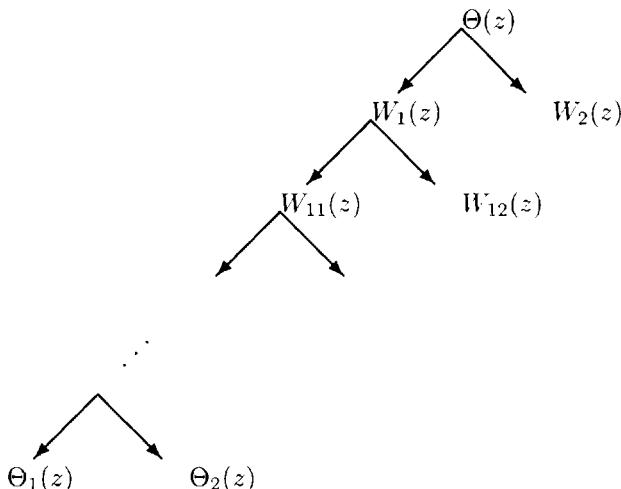


FIGURE 5. *Divide-and-conquer approach to factorization of rational matrix functions.*

### 13. Acknowledgement

The author is thankful to Richard Brent, Harry Dym, Israel Gohberg, Gene Golub, Ming Gu, Georg Heinig, Thomas Kailath, Michel Stewart, and Marc Van Barel for valuable discussions and suggestions.

### References

- [A65] N.I.Akhiezer, *The classical moment problem and some related problems in analysis*. Hafner Publishing Co., New York, 1965.
- [AD86] D.Alpay and H.Dym, *On applications of reproducing kernel spaces to the Schur algorithm and rational J-unitary factorizations*, in I.Schur Methods in Operator Theory and Signal Processing (I.Gohberg, ed.), OT18, Birkhäuser Verlag, Basel, 1986, 89–160.
- [AG88] D.Alpay and I.Gohberg, *Unitary rational matrix functions* in Topics in interpolation theory of rational matrix functions (I.Gohberg, ed.), OT33, Birkhäuser Verlag, Basel, 1988, 175–222.

- [AD93] D.Alpay and H.Dym, *On a new class of reproducing kernel spaces and a new generalization of the Iohvodov laws*, Linear Algebra and Its Applications, **178** (1993), 162–179.
- [AG90] G.Ammar and P.Gader, *New decompositions of the inverse of a Toeplitz matrix*, Signal processing, Scattering and Operator Theory, and Numerical Methods, Proc. Int. Symp. MTNS-89, vol. III, 421-428, Birkhauser, Boston, 1990.
- [B37] J.P.M. Binet, *Observations sur des théorèmes de Géométrie, énoncées page 160 de ce volume et page 222 du volume et page 222 du volume précédent*, Journ. (de Liouville) de Math, ii. (1837), 248-252.
- [B71] J.R.Bunch, *Analysis of the diagonal pivoting method*, SIAM J. Num. Anal., **8** (1971), 656 -680.
- [BA80] R. Bitmead and B. Anderson, *Asymptotically fast solution of Toeplitz and related systems of linear equations*, Linear Algebra and its Applications, **34** (1980), 103–116.
- [BBHS95] A.W.Bojanczyk, R.P.Brent, F.R. de Hoog and D.R.Sweet, *On the stability of the Bareiss and related Toeplitz factorization algorithms*, SIAM J. on Matrix Analysis Appl., **16**No.1 (1995).
- [BKO95a] T.Boros, T.Kailath and V.Olshevsky. *The fast Björck-Pereyra-type algorithm for solving Cauchy linear equations*, submitted, 1995.
- [BKO95b] T.Boros, T.Kailath and V.Olshevsky, *Predictive pivoting and backward stability of fast Cauchy solvers*, 1995, submitted.
- [BGK79] H. Bart, I. Gohberg and M.A.Kaashoek, *Minimal factorization of matrix and operator functions*, OT1, Birkhäuser Verlag, Basel, 1979.
- [BGKV80] H.Bart, I.Gohberg, M.A.Kaashoek and P.Van Dooren, *Factorizations of transfer functions*, SIAM J. Control and Optim., **18** (1980), 675-696.
- [BGR90] J. Ball, I.Gohberg and L.Rodman, *Interpolation of rational matrix functions*, OT45, Birkhäuser Verlag, Basel, 1990.
- [BK77] J.R.Bunch and L.Kaufman, *Some stable methods for calculating inertia and solving symmetric linear systems*, Math. Comp., **31**, 162 - 179.
- [BP70] A.Björck and V.Pereyra, *Solution of Vandermonde Systems of Equations*, Math. Comp., **24** (1970), 893-903.
- [BP94] D.Bini and V.Pan, *Polynomial and Matrix Computations*, Volume 1, Birkhauser, Boston, 1994.
- [C41] A.L. Cauchy, *Mémoire sur les fonctions alternées et sur les sommes alternées*, Exercices d'analyse et de phys. math., ii (1841), 151-159.
- [C80] G.Cybenko, *The numerical stability of Levinson-Durbin algorithm for Toeplitz systems of equations*, SIAM J. Sci. Statist. Comput., **1** (1980), 303 – 319.
- [C88] T.Chan, *An optimal circulant preconditioner for Toeplitz systems*, SIAM J. Sci. Stat. Comput. **9** (4) (1988), 166 - 771.
- [C89] J.Chun, *Fast array algorithms for structured matrices*, Ph.D.Thesis, 1989, Stanford University, Stanford, CA.
- [CF88] T.Chan and D.Foulser, *Effectively well-conditioned linear systems*, SIAM J. Sci. Stat. Computations, 9 (1988), 963 – 969.
- [CH92] T.Chan and P.Hansen, *A look-ahead Levinson algorithm for indefinite l Toeplitz systems*, SIAM J. on Matrix Anal. and Appl., 13(1992), 1079-1090.
- [CKLA87] J.Chun, T.Kailath and H.Lev-Ari, *Fast parallel algorithms for QR and triangular factorizations*, J. Sci. Stat. Comput., **8**(1987), 899-913.
- [CK91] J.Chun and T.Kailath, *Fast triangularization and orthogonalization of Hankel and Vandermonde matrices*, Linear Algebra Appl., **151** (1991), 199 – 228.
- [CR93] D.Calvetti and L.Reichel, : *Fast inversion of Vandermonde-like matrices involving orthogonal polynomials*, BIT, 1993.
- [D82] J.M.Delosme, *Fast algorithms for finite shift-rank processes*, Ph.D.Thesis, Stanford University, Stanford, CA, 1982.
- [DGK81] Ph.Delsarte, Y.Genin and Y.Kamp, *On the role of the Nevanlinna-Pick problem in circuit and system theory*, Circuit Theory and Appl., **9** (1981), 177-187.
- [DBP77] C. de Boor and A. Pinkus, *Backward error analysis for totally positive linear systems*, Numer. Math., **27** (1977), 485-490.
- [F84] M.Fiedler, *Hankel and Loewner matrices*, Linear Algebra and Appl., **28**(1984), 75-95.

- [FMKL79] B.Friedlander, M.Morf, T.Kailath and L.Ljung, *New inversion formulas for matrices classified in terms of their distance from Toeplitz matrices.* Linear Algebra and Appl., **27** (1979), 31-60.
- [FZ93] R.Freund and H.Zha, *A look-ahead strategy for the solution of general Hankel systems.* Numerische Mathematik, **64**, (1993), 295-322.
- [G95] Ming Gu, *Stable and efficient algorithms for structured systems of linear equations.* to appear in SIAM J. on Matrix Analysis and Appl., 1997.
- [GH94] M.Gutknecht and M.Hochbruck, *Look-ahead Levinson and Schur recurrences in the Pad'e table,* Electronic Transactions on Numerical Analysis, **2** (1994). 104-129.
- [GK50] F.R.Gantmacher and M.G.Krein, *Oscillatory matrices and kernels, and small vibrations of mechanical systems,* second edition, (in Russian). GITTL. Moscow, 1950. *German translation : Oszillationsmatrizen. Oszillationskerne und kleine Schwingungen mechanischer Systeme.* Berlin, Akademie Verlag, 1960.
- [GK89] I.Gohberg and I.Koltracht. *Efficient algorithm for Toeplitz plus Hankel matrices.* Integral Equations and Operator Theory, **12** (1989). 136 – 142.
- [GK93] I.Gohberg and I.Koltracht. *Mixed, componentwise and structured condition numbers.* SIAM J. Matrix Anal. Appl., **14**(1993). 688–704.
- [GKKL87] I.Gohberg, T.Kailath, I.Koltracht and P.Lancaster. *Linear Complexity parallel algorithms for linear systems of equations with recursive structure.* Linear Algebra Appl., **88/89** (1987), 271–315.
- [GKO95] I.Gohberg, T.Kailath and V.Olshevsky, *Fast Gaussian elimination with partial pivoting for matrices with displacement structure.* Math. of Comp., **64** (1995). 1557-1576.
- [GLR82] I.Gohberg, P.Lancaster and L.Rodman. *Matrix Polynomials.* Academic Press. New York, 1982.
- [GO92] I.Gohberg and V.Olshevsky. *Circulants, displacements and decompositions of matrices,* Integral Equations and Operator Theory, **15**, No. 5 (1992). 730 -743.
- [GO93] I.Gohberg and V.Olshevsky, *Fast algorithm for matrix Nehari problem.* Proceedings of MTNS-93, Systems and Networks: Mathematical Theory and Applications. v.2. Invited and Contributed Papers,edited by U. Helmke, R. Mennicken and J. Sauers. Academy Verlag, 1994, p. 687-690.
- [GO94a] I.Gohberg and V.Olshevsky. *Fast inversion of Chebyshev-Vandermonde matrices.* Numerische Mathematik, **67**, No. 1 (1994). 71 – 92.
- [GO94b] I.Gohberg and V.Olshevsky, *Fast state space algorithms for matrix Nehari and Nehari-Takagi interpolation problems,* Integral Equations and Operator Theory, **20**. No. 1 (1994), 44 – 83.
- [GO94c] I.Gohberg and V.Olshevsky, *Complexity of multiplication with vectors for structured matrices,* Linear Algebra Appl., **202** (1994). 163 – 192.
- [GO97] I.Gohberg and V.Olshevsky, *The fast generalized Parker-Traub algorithm for inversion of Vandermonde and related matrices,* J.of Complexity, 1997.
- [GTVDV96] K.Gallivan, S.Thirumalai, P.Van Dooren, V.Vermaut. *High performance algorithms for Toeplitz and block Toeplitz matrices.* Linear Algebra Appl., **241-243**(1996). 343-388.
- [GP92] M.Gasca, J.M.Pena, *Total positivity and Neville elimination.* Linear Algebra Appl. **165**(1992), 25-44.
- [GVKDM83] Y.Genin, P. Van Dooren, T.Kailath, J.Delsome and M.Morf. *On  $\Sigma$ -lossless transfer functions and related questions,* Linear Algebra Appl., **50** (1983). 251 - 275.
- [GVL96] G. Golub and C. Van Loan, *Matrix Computations.* second edition. John Hopkins U. P., Baltimore, 1996.
- [H95] G. Heinig, *Inversion of generalized Cauchy matrices and other classes of structured matrices,* in Linear Algebra in Signal Processing. IMA volumes in Mathematics and its Applications, vol. **69** (1995), 95 - 114.
- [HB97] G.Heinig and A.Bojanczyk, *Transformation techniques for Toeplitz and Toeplitz-plus-Hankel matrices, I. Transformations,* to appear in Linear Algebra Appl.. 1997.
- [HJR88] G.Heinig, P.Jankowski and K.Rost, *Fast inversion of Toeplitz-plus-Hankel matrices.* Numer.Math. **52** (1988), 665 – 682.
- [HLM95] C.He, A.Laub and V.Mehrmann. *Placing plenty of poles is pretty preposterous.* preprint, 1995.

- [HO01] G. Heinig and V.Olshevsky, *The Schur algorithm for matrices with Hessenberg displacement structure*, in *Structured Matrices in Mathematics, Computer Science and engineering*, vol. II, Contemporary Mathematics, CONM 281, p. 3–16, AMS Publications, Providence, 2001.
- [HR84] Heinig G., Rost K., *Algebraic methods for Toeplitz-like matrices and operators*, Operator Theory, vol. 13, Birkhauser, Basel, 1984.
- [Hig87] N.J.Higham, Error analysis of the Björck-Pereyra algorithms for solving Vandermonde systems, *Numer. Math.*, 50 (1987), 613 – 632.
- [Hig88] N.J.Higham, Fast solution of Vandermonde-like systems, involving orthogonal polynomials, *IMA J. Numer. Anal.*, 8 (1988), 473-486.
- [Hig90] N.J.Higham, Stability analysis of algorithms for solving confluent Vandermonde-like systems, *SIAM J. Matrix Anal. Appl.*, 11(1) (1990), 23–41.
- [Hig95] N.J.Higham, *Stability of the diagonal pivoting method with partial pivoting*, Preprint, 1995.
- [Hig96] N.J.Higham, *Accuracy and Stability of Numerical Algorithms*, SIAM, Philadelphia, 1996.
- [K72] S.Karlin, *Total Positivity*, vol 1, Stanford University Press, Stanford, 1972.
- [K80] W.Kahan, Interval arithmetic options in the proposed IEEE floating point arithmetic standard, in *Interval Mathematics*, 1980, (Karl L.E.Nickel, editor), Academic Press, New York, 1980, 99-128.
- [K86] T.Kailath, A theorem of I.Schur and its impact on modern signal processing, in *Operator Theory: Advances and Applications* (I.Schur methods in Operator Theory and Signal Processing), 18, 9-30, Birkhauser, 1986.
- [K87] T.Kailath, Signal processing applications of some moment problems, Proc. of Symposia in Appl. Math., vol. 37, 71-109. AMS annual meeting, short course reprinted in *Moments in mathematics*, ed. H.Landau, San Antonio, TX, January 1987.
- [KL96] Misha E. Kilmer and Dianne P. O’Leary, *Pivoted Cauchy-Like Preconditioners for Regularized Solution of Ill-Posed Problems*, Computer Science Department Report CS-TR-3682, Institute for Advanced Computer Studies Report UMIACS-TR-96-63, University of Maryland, September 1996.
- [K97] Misha E. Kilmer, *Cauchy-Like Preconditioners for 2-Dimensional Ill-Posed Problems*, Computer Science Department Report CS-TR-3776, University of Maryland, March 1997.
- [K99] T. Kailath, *Displacement structure and array algorithms*, In: *Fast reliable algorithms for matrices with structure*, (T. KAILATH, A.H. SAYED, Eds.),SIAM Publications, Philadelphia 1999.
- [KKM79] T.Kailath, S.Kung and M.Morf, Displacement ranks of matrices and linear equations, *J. Math. Anal. and Appl.*, 68 (1979), 395-407.
- [KN36] M.G.Krein and M.A.Naimark, *The Method of Symmetric and Hermitian Forms in the Theory of Separation of the Roots of Algebraic Equations*, Linear and Multilinear Algebra, 1981, 10, 265-308. (originally published in Russian in Kharkov, 1936).
- [KO95] T.Kailath and V.Olshevsky, *Displacement structure approach to Chebyshev-Vandermonde and related matrices*, Integral Equations and Operator Theory, 1995.
- [KO96] T.Kailath and V.Olshevsky, *Displacement structure approach to discrete-trigonometric-transform based preconditioners of G.Strang and T.Chan types*, submitted, 1996; a short 15 pages version will appear in the Italian journal Calcolo, the issue devoted to the Proc. of the international workshop on Toeplitz matrices, Cortona, Italy, September 1996.
- [KO96b] T.Kailath and V.Olshevsky, *Unitary Hessenberg matrices, and the generalized Parker-Forney-Traub and Björck-Pereyra algorithms for Szegö-Vandermonde matrices*, ISL technical report, 1996.
- [KO97a] T.Kailath and V.Olshevsky, *Displacement structure approach to polynomial Vandermonde and related matrices*, preprint, 1994, to appear in *Linear Algebra Appl.*, 1997.
- [KO97b] T.Kailath and V.Olshevsky, *Bunch-Kaufman Pivoting for Partially Reconstructible Cauchy-like Matrices, with Applications to Toeplitz-like Linear Equations and to Boundary Rational Matrix Interpolation Problems*, to appear in *Linear Algebra Appl.*, the issue devoted to Proc. of ILAS-95 symposium.

- [KS92] T. Kailath and A.H.Sayed, *Fast algorithms for generalized displacement strcutures.* in Recent advances in mathematical theory of systems, control, networks and signal processing II, Proc. of the MTNS-91 (H.Kimura, S.Kodama. Eds.). Mita Press. Japan. 1992, 27 – 32.
- [KS95] T.Kailath and A.H.Sayed, *Displacement structure : Theory and Applications.* SIAM Review, **37** No.3 (1995), 297-386.
- [KS95b] T.Kailath and A.H.Sayed. *A look-ahead block Schur algorithm for Toeplitz-like matrices.* SIAM J. of Matrix Analysis Appl., **16** (1995), 388-414.
- [KVB96] P.Kravanija and M. Van Barel, *A fast Hankel solver based on inversion formula for Loewner matrices.* preprint. 1996.
- [L74] F. Lander. *The Bezoutian and the inversion of Hankel and Toeplitz matrices* (in Russian). Matem. Issled., Kishinev. **9** (No. 2). 69 - 87.
- [LAPACK] E.Anderson, Z.Bai, C.Bisoff, J.Demmel, J.Dongarra, J. Du Croz, A.Greenbaum, S.Hammarling, A.McKenney, S.Ostrouchov and D.Sorensen. *LAPACK User's Guide. Release 2.0.* SIAM. Philadelphia, PA. USA. second edition. 1995.
- [LA83] H.Lev-Ari. *Nonstationary Lattice-Filter Modeling.* Ph.D. thesis. Stanford University. December 1983.
- [LA96] H.Lev-Ari. *Displacement structure: two related perspectives.* pp. 233-242. in Communications, Computation, Control and Signal Processing, a tribute to Thomas Kailath (edited by A.Paulraj, V.Raychowdhury and C.Schaper). Kluwer Academic Publishers. Boston, 1996.
- [LAK84] H.Lev-Ari and T.Kailath. *Lattice filter parameterization and modeling of nonstationary processes.* IEEE Trans on Information Theory. **30** (1984), 2-16.
- [LAK86] H.Lev-Ari and T.Kailath. *Triangular factorization of structured Hermitian matrices.* in *Operator Theory : Advances and Applications* (I.Gohberg, ed.), vol. **18**. 301 - 324. Birkhäuser, Boston, 1986.
- [LAK92] H.Lev-Ari and T.Kailath *State-space approach to factorization of lossless transfer functions and structured matrices.* Linear Algebra Appl., **162 - 164** (1992), 273 - 295.
- [M74] M.Morf. *Fast algorithms for Multivariable Systems.* Ph.D.Thesis. Stanford University. Stanford, CA, 1974.
- [M80] M.Morf. *Doubling algorithms for Toeplitz and related equations.* Proc. IEEE Inter. Conf. on Acoustics, Speech and Signal Processing. Denver. 1980. 954-959.
- [MS77] F.J. McWilliams and N.J.Sloane. *The Theory of Error-Correcting Codes.* North-Holland Publishing Company. Amsterdam. 1977.
- [MB79] J.Maroulas and S.Barnett, Polynomials with respect to a general basis. I. Theory. *J. of Math. Analysis and Appl.*, **72** : 177 -194 (1979).
- [N19] R.Nevanlinna, Über beschränkte funktionen die in gegebenen punkten vorgeschreibene funktionswerte bewirkt werden. *Anal. Acad. Sci. Fenn.*, **13** (1919), 1 -71.
- [O93] M.Ohsmann. *Fast cosine transform of Toeplitz matrices. algorithm and applications.* IEEE Transactions on Signal Processing, **41**, No. **10** (1993), 3057-3061.
- [O95] M.Ohsmann. *Fast transforms of Toeplitz matrices.* Linear Algebra Appl., **231** (1995), 181-192.
- [OO02] A.Olshevsky and V.Olshevsky. *Several matrix interpretations of the Euclid algorithm and its generalizations.* in preparation. 2002.
- [OP98] V. Olshevsky and V. Pan. “A superfast state-space algorithm for tangential Nevanlinna-Pick interpolation problem.” in *Proceedings of the 39th IEEE Symposium on Foundations of Computer Science*. pp. 192–201. 1998.
- [OS99] V. Olshevsky and A. Shokrollahi, “A Displacement Approach to Efficient Decoding of Algebraic Codes.” in *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, 235–244, 1999.
- [OS99a] V. Olshevsky and M.Stewart. “Stable Factorizations of Hankel and Hankel-like matrices.” in *Advanced Signal Processing Algorithms, Architectures, and Implementations.*, 334–349. SPIE Publications. 1999.
- [OS01] V.Olshevsky and A.Shokrollahi. *A superfast algorithm for confluent rational tangential interpolation problem via matrix-vector multiplication for confluent Cauchy-like*

*matrices*, in *Structured Matrices in Mathematics, Computer Science and engineering*, vol. I, Contemporary Mathematics, CONM 280, p. 31–46, AMS Publications, Providence, 2001.

- [OS01a] V. Olshevsky and M. Stewart, “Stable Factorizations of Hankel and Hankel-like matrices,” Numerical Linear Algebra, **8**, Issue 6/7 (2001), 401–434.
- [OS86] A.Odlyzko and A.Schönhage, *Fast algorithms for multiple evaluations of the Riemann zeta function*, Technical report, AT&T Bell Labs, Murray Hill, N.J., 1986.
- [P60] Y.Potapov, *The multiplicative structure of  $J$ -contractive matrix functions*, Amer. Math.Translations, **15** (1960) 131–244.
- [P90] V.Pan, *On computations with dense structured matrices*, Math. of Computation, **55**, No. 191 (1990), 179 – 190.
- [R90a] L.Reichel, *A matrix problem with application to rapid solution of integral equations*, SIAM J. Sci. Stat. Comput, **11** (1990), 263–280.
- [R90b] L.Reichel, *Newton interpolation at Leja points*, BIT, 30(1990), 23 – 41.
- [Ro85] V. Rokhlin, *Rapid solution of integral equations of classic potential theory*, J. Comput Phys. **60** (1985), 187–207.
- [RO91] L. Reichel and G. Opfer, *Chebyshev-Vandermonde Systems*, Math. of Comp., **57**(1991), 703–721.
- [RS94] Reed-Solomon codes and their applications, (S.Wicker and V.Bhargava, eds.), IEEE Press, New York, 1994.
- [S17] I.Schur, *Über potenzreihen die im Inneren des Einheitskreises beschränkt sind*, Journal für die Reine und Angewandte Mathematik, **147** (1917), 205 – 232. English translation in *Operator Theory : Advances and Applications* (I.Gohberg, ed.), vol. **18**, 31 – 88, Birkhäuser, Boston, 1986.
- [S86] L.Sakhnovich, *Factorization problems and operator identities*, Russian Mathematical Surveys, **41** : 1 (1986), 1 - 64.
- [S92] A.H.Sayed, *Displacement structure in signal processing in engineering and mathematics*, Ph.D.Thesis, 1992, Stanford University, Stanford, CA.
- [S96] M.Stewart, *Pivoting for Stability in the fast factorization of Cauchy-like matrices*, preprint, 1996.
- [S97] S. Serra Cappizzano, *A Korovkin-type theory, for finite Toeplitz operators via matrix algebras*, preprint, 1997.
- [SB80] J.Stoer and R.Bulirsch, *Introduction to Numerical Analysis*, Springer-Verlag, New York, 1980.
- [SB95] D.Sweet and R.Brent, *Error analysis of a partial pivoting method for structured matrices*, Advanced Signal processing algorithms, Proc of SPIE-1995, vol. **2563**, 266–280.
- [SKLAC94] A.Sayed,T.Kailath, H.Lev-Ari and T.Constantinescu, *Recursive solutions of rational interpolation problems via fast matrix factorization*, Integral Equations and Operator Theory, 1994.
- [SLAK93] A.Sayed, H.Lev-Ari and T.Kailath, *Fast triangular factorization of the sum of quasi-Toeplitz and quasi-Hankel matrices*, Linear Algebra and Appl., **191** (1993), 77 – 106.
- [T86] M.Trummer, *An efficient implementation of conformal mapping method using the Szegő kernel*, SIAM J. Numer. Anal., **23** (1986), 853-872.
- [TG81] W.Tang and G.Golub, *The block decomposition of a Vandermonde matrix and its applications*, BIT, **21** (1981), 505-517.
- [T94] E. Tyrtyshnikov, *How bad are Hankel matrices?* Numerische Mathematik, **67** (1994), 261–269.
- [V93] J. M. Varah, *Errors and Perturbations in Vandermonde Systems*, IMA Journal of Numer. Anal., **13** (1993), 1-12.
- [W68] J.H.Wilkinson, *A priori error analysis of algebraic processes*, Proc. Intern. Congr. Math. (1966), pp. 629-639, Moskow, Mir 1968.
- [W65] J.H.Wilkinson, *The algebraic eigenvalue problem*, Clarendon Press, Oxford, 1965.
- [W71] J.H.Wilkinson, *Modern error analysis*, SIAM Review, **14**(1971), 548-568.

*This page intentionally left blank*

# Inversion of Toeplitz-plus-Hankel matrices with arbitrary rank profile

Georg Heinig

**ABSTRACT.** Fast algorithms for the inversion of Toeplitz-plus-Hankel matrices are presented that work, in contrast to algorithms in the literature, for any nonsingular Toeplitz-plus-Hankel matrix. The design of the algorithms is based on the kernel structure properties of Toeplitz-plus-Hankel matrices.

## 1. Introduction

This paper deals with fast algorithms for the solution of linear systems with a nonsingular  $n \times n$  Toeplitz-plus-Hankel coefficient matrix (T+H-matrix), i.e. a matrix  $R_n = [a_{ij}]_{i,j=1}^n$  with  $a_{ij} = c_{i-j} + b_{i+j-1}$ . The entries  $a_{ij}$  are from a given field  $\mathbb{F}$ . If not stated otherwise, we assume that the characteristic of  $\mathbb{F}$  is not equal to 2. At the end of some sections the case of characteristic 2 will briefly be discussed. “Fast” means that the computational complexity is only  $O(n^2)$ , compared with  $O(n^3)$  for a general system.

Algorithms with this amount have been designed in many papers like [24], [25], [14], [16], [5], [27], [28], [29], [15], [23] and others. However, all algorithms in the literature do not work for arbitrary nonsingular T+H-matrices. In many cases the algorithms do not work even for all positive definite matrices. Levinson-type algorithms under the only assumption of strong nonsingularity have been presented, for example, in [14] and [5], Schur-type algorithms in [29] and [15]. Recall that strong nonsingularity of a matrix means the nonsingularity of all leading principal submatrices  $[a_{ij}]_{i,j=1}^k$  ( $k = 1, \dots, n$ ), so that the positive definite case is included. In the papers [28] and [23] algorithms have been designed under the condition of nonsingularity of all central submatrices  $[a_{ij}]_{i,j=l}^{n+1-l}$ . This also includes the positive definite case.

In case that the entries of the matrix are real or complex numbers there is a way how additional assumptions can be avoided. This way is to transform the matrix into a Cauchy-like matrix with the help of discrete Fourier or real trigonometric transformations (see [9], [4], [12]). The advantage of the Cauchy-like structure is that permutations of rows and columns do not destroy the structure, so that break downs can be avoided by applying pivoting.

---

1991 *Mathematics Subject Classification*. Primary: 15A23, 15A57, Secondary: 65F05.

The work was supported by Research Grant SM 05/02 of Kuwait University.

However, if the entries of the matrix are elements of a field in which a transformation like the DFT is not available, then the question remains how to find the inverse of a T+H-matrix with the help of a fast algorithm. There is another motivation for designing fast algorithms without transformations, namely the following. For the transformation approach one needs the complete matrix from the very beginning. However, in many applications the entries of the matrix come in successively, so the main problem is to update a solution using the new incoming data.

Let us explain the basic ideas of our approach. The approach relies, firstly, on the fact that the inverse of a T+H-matrix can be represented in terms of a “generating system” of the inverse matrix, which consists of 4 vectors, which can be interpreted in different way such as solutions of certain “fundamental equations”, as the columns of the inverse of the matrix and an extension, or as solutions of a related homogeneous system (see [14], [23]). The “classical” algorithms construct recursively the generating systems for subsequent nested nonsingular submatrices of the T+H-matrix.

The question is what to do if some of these submatrices are singular. One idea is to jump in a look-ahead manner from one nonsingular section to the next one. This works fine for scalar Toeplitz and Hankel matrices (see [16]), but is not appropriate for block Hankel and Toeplitz matrices, and seemingly also not for T+H-matrices. The reason is that it may happen that there are not enough nonsingular submatrices, so that an overall complexity of  $O(n^2)$  cannot be guaranteed.

A second idea, which leads to the approach in this paper, is to utilize the kernel structure properties of T+H-matrices. This was worked out for scalar Hankel matrices (see [13]), for block Hankel matrices (see [8]) and also for other classes like generalized Cauchy-Vandermonde matrices (see [10]).

The kernel structure of T+H-matrices was studied in the recent paper [11]. The main result of this paper is that the kernel of T+H-matrices can be expressed in terms of a *fundamental system* of this matrix, which is a system of 4 vectors. Let us point out that the concept of a fundamental system is defined for any T+H-matrix, in particular for a nonsingular T+H-matrix. It actually does not depend on the size of the matrix but only on the data from which the matrix is built. The fundamental system of a T+H-matrix indicates in particular whether the matrix is nonsingular, and a generating system for the inverse can easily be obtained.

The rest of the present paper is built as follows. In Section 2 we discuss briefly a formula for the inverse of a T+H-matrix, which was presented [17] and [14], that can be used for fast solution of T+H-systems of equations. Note that a modification of the formula is discussed in [23]. In Section 3 we quote and discuss the main result of [11] concerning the kernel structure of T+H-matrices and introduce the concept of a fundamental system which is basic for our approach. In Section 4 we study how the fundamental systems changes after the extension of the matrix by a row. Note that the extension by two rows, one at the top and one at the bottom, is equivalent to the extension by two columns, one at the left and one at the right. That means the recursion provides an algorithm for finding the fundamental system of any T+H-matrix. This algorithm will be presented in Section 6.

Unlike the classical algorithms, the algorithm will not provide a triangular factorization. If one is interested in such a factorization one has to study also single column extension. This can be done via column deletion followed by double

column extension. For this reason we study the change of a fundamental system after column deletion in Section 5.

The algorithm presented in Section 6 is a Levinson-type algorithm since it computes the residuals via inner products. To avoid inner product calculations, which is a bottleneck in parallel computations, one can precompute the residuals by a Schur-type algorithm. This will be shown in Section 7. It will also be shown that a combination of the Levinson-type and Schur-type algorithm can be used to design (at least in principle) an algorithm with complexity less than  $O(N^2)$ , provided that a multiplication of polynomials of degree  $n$  with coefficients in  $\mathbb{F}$  with a complexity less than  $O(n^2)$  is possible.

**Notations.** If  $x = (x_j)_{j=1}^n \in \mathbb{F}^n$ , then  $x(t)$  will denote the polynomial  $x(t) = \sum_{j=1}^n x_j t^{j-1}$ . Let  $J_k$  denote the  $k \times k$  matrix of the counteridentity

$$J_k = \begin{bmatrix} 0 & & 1 \\ & \ddots & \\ 1 & & 0 \end{bmatrix} \} k .$$

A vector  $x \in \mathbb{F}^k$  is called *symmetric* if  $x = J_k x$ , it is called *skewsymmetric* if  $x = -J_k x$ . The subspace of all symmetric vectors in  $\mathbb{F}^n$  will be denoted by  $\mathbb{F}_+^k$  or  $\mathbb{F}_1^k$  and the subspace of all skewsymmetric vectors by  $\mathbb{F}_-^k$  or  $\mathbb{F}_{-1}^k$ . Note that if  $\mathbb{F}$  has characteristic 2, then the subspaces  $\mathbb{F}_+^k$  and  $\mathbb{F}_-^k$  coincide.

A sequence  $\mathbf{a} = (a_1, \dots, a_N)$ ,  $a_j \in \mathbb{F}$ , will be associated with the family of  $l \times k$  Hankel matrices,  $k = 1, \dots, N$ ,  $k + l = N + 1$ .

$$(1.1) \quad H_k(\mathbf{a}) = \begin{bmatrix} a_1 & a_2 & a_3 & \cdots & a_k \\ a_2 & a_3 & & \ddots & a_{k+1} \\ a_2 & & & & \\ \vdots & \ddots & & \ddots & \vdots \\ a_l & a_{l+1} & \cdots & & a_N \end{bmatrix} .$$

Since an  $l \times k$  matrix  $T$  is Toeplitz if and only if  $H = TJ_k$  is Hankel and  $T = HJ_k$ , we can represent  $l \times k$  T+H-matrices in the form

$$A_k(\mathbf{a}, \mathbf{b}) = H_k(\mathbf{a}) + H_k(\mathbf{b})J_k.$$

Note that this representation is not unique, since the classes of  $l \times k$  Toeplitz and Hankel matrices have a nontrivial, 2-dimensional (for  $k, l > 1$ ) intersection. Besides  $A_k(\mathbf{a}, \mathbf{b})$  we consider the matrices

$$A_k^-(\mathbf{a}, \mathbf{b}) = H_k(\mathbf{a}) - H_k(\mathbf{b})J_k.$$

For convenience we denote  $A_k^{(1)}(\mathbf{a}, \mathbf{b}) = A(\mathbf{a}, \mathbf{b})$  and  $A_k^{(-1)}(\mathbf{a}, \mathbf{b}) = A_k^-(\mathbf{a}, \mathbf{b})$ .

## 2. Formula for the inverse

In this section we present some results from [17] and [14]. Throughout the section, we assume that  $A_n(\mathbf{a}, \mathbf{b})$  is a nonsingular T+H matrix ( $n \geq 2$ ).

We introduce a transformation  $\nabla$  defined for matrices  $B = [b_{ij}]_{i,j=1}^n$  by

$$\nabla(B) = [\tilde{b}_{ij}]_{i,j=0}^{n+1}, \quad \tilde{b}_{ij} = b_{i-1,j} + b_{i+1,j} - b_{i,j-1} - b_{i,j+1},$$

in which we set  $b_{ij} = 0$  if  $i \notin \{1, \dots, n\}$  or  $j \notin \{1, \dots, n\}$ .

This transformation can be written in a useful and nice form in terms of the generating function of a matrix, which is defined as follows. If  $B = [b_{ij}]_{i,j=1}^n$  then  $B(t, s)$  will be the bivariate polynomial

$$B(t, s) = \sum_{i,j=1}^n b_{ij} t^{i-1} s^{j-1}.$$

It is easily checked that

$$\nabla(B)(t, s) = (ts - 1)(t - s)B(t, s).$$

The following fact is proved in [17].

**PROPOSITION 2.1.** *The  $n \times n$  matrix  $B$  is the inverse of a nonsingular  $T+H$ -matrix  $A_n(\mathbf{a}, \mathbf{b})$  if and only if  $\text{rank } \nabla(B) = 4$ . Furthermore, the range of  $\nabla(B)$  is equal to the kernel of  $A_{n+2}(\mathbf{a}, \mathbf{b})$ .*

Any basis of the kernel of  $A_{n+2}(\mathbf{a}, \mathbf{b})$  will be called *generating system*<sup>1</sup> for  $A_n(\mathbf{a}, \mathbf{b})^{-1}$ , since this matrix can be constructed from it. In this paper we present only the explicit “Bezoutian” representation of the inverse that includes also a generating system for the transpose of  $A_n(\mathbf{a}, \mathbf{b})^{-1}$ , which is a generating system for  $A_n(\mathbf{a}, J_N \mathbf{b})$ . Concerning a representation without the generating system of the transpose, which is however less efficient for the solution of linear systems, we refer to [14]).

In order to get a representation for  $\nabla(A_n(\mathbf{a}, \mathbf{b})^{-1})$ , we select among the generating systems a “canonical” one. For this we introduce a matrix  $F$  of test functionals by

$$F = \begin{bmatrix} a_{-1} + b_{n+2} & a_0 + b_{n+1} & \dots & a_{n+1} + b_0 & a_{n+2} + b_{-1} \\ 1 & 0 & \dots & 0 & 0 \\ a_n + b_{2n+1} & a_{n+1} + b_{2n} & \dots & a_{2n} + b_{n+1} & a_{2n+1} + b_n \\ 0 & 0 & \dots & 0 & 1 \end{bmatrix}.$$

Here  $a_{-1}, a_0, a_{2n}, a_{2n+1}$  and  $b_{-1}, b_0, b_{2n}, b_{2n+1}$  are arbitrary. A generating system  $\{x_1, x_2, x_3, x_4\}$  will be called *canonical* if  $F[x_1 \ x_2 \ x_3 \ x_4] = I_4$ . If the generating system is canonical, then, in particular, the first and last components of  $x_1$  and  $x_3$  vanish, and if these components are deleted one obtains the first and last column of  $A_n(\mathbf{a}, \mathbf{b})^{-1}$ .

Any given generating system can be easily transformed into a canonical one with  $O(n)$  operations. In fact, if  $\{x_1, x_2, x_3, x_4\}$  is any fundamental system, then  $\Theta = F[x_1 \ x_2 \ x_3 \ x_4]$  is nonsingular and the columns of  $[x_1 \ x_2 \ x_3 \ x_4] \Theta^{-1}$  form a canonical fundamental system.

**THEOREM 2.2.** [17], [14] *Let  $(x_j)_{j=1}^4$  be a canonical generating system for  $A_n(\mathbf{a}, \mathbf{b})^{-1}$  and  $(\tilde{x}_j)_{j=1}^4$  a canonical generating system for its transpose. Then*

$$A_n(\mathbf{a}, \mathbf{b})^{-1}(t, s) = \frac{x_2(t)\tilde{x}_1(s) - x_1(t)\tilde{x}_2(s) + x_4(t)\tilde{x}_3(s) - x_3(t)\tilde{x}_4(s)}{(t - s)(ts - 1)}.$$

Theorem 2.2 can be used in different ways. Firstly, using original definition of the transformation  $\nabla$  the inverse matrix can be constructed recursively, column by column, in  $O(n^2)$  operations, from a generating system for  $A_n(\mathbf{a}, \mathbf{b})^{-1}$  and its transpose. Secondly, if  $\mathbb{F}$  is the field of complex numbers, then the formula for the

---

<sup>1</sup>In other papers such a system is referred to a *fundamental system*. However, we use this concept in another (but related) sense.

inverse matrix can be written in matrix form containing only DFT's and diagonal matrices (see [19]). Thus matrix-vector multiplication, i.e. the solution of a linear system, can be carried out with  $O(n \log n)$  computational complexity. If  $\mathbb{F}$  is the field of reals, then this can be done with the help of the Discrete Hartley transformation (see [20]) or with the help of sine and cosine transformations (see [21]). For some finite fields  $\mathbb{F}$  number theoretic transformations can be used. Finally, the third possibility is to use a transformation free matrix representations which can be applied in any field. One of them can be found in [18]. For some more information about calculations with Toeplitz-plus-Hankel-like matrices we refer to the monograph [1].

### 3. Kernel structure and fundamental systems

In the previous section we have shown that the inverse of a T+H matrix  $A_n(\mathbf{a}, \mathbf{b})$  can be constructed from bases of the 4-dimensional kernels of the  $(n-2) \times (n+2)$  matrices  $A_{n+2}(\mathbf{a}, \mathbf{b})$  and  $A_{n+2}(\mathbf{a}, J_N \mathbf{b})$ . This makes it important to study its evolution after modifying, in particular extending, the vectors  $\mathbf{a}$  and  $\mathbf{b}$ . However, if the T+H-matrix of the modified vectors is singular, then this kernel is not anymore 4-dimensional. Therefore, it is desirable to study the structure of the kernels of the whole family of matrices  $A_k(\mathbf{a}, \mathbf{b})$ . This is not enough: One has also to combine it with the investigation of the kernels of the matrices  $A_k^-(\mathbf{a}, \mathbf{b})$ .

The fact that there is some relation between a Toeplitz-plus-Hankel and the related Toeplitz-minus-Hankel matrix was already observed in [7]. It turned out that the kernels of both families  $A_k(\mathbf{a}, \mathbf{b})$  and  $A_k^-(\mathbf{a}, \mathbf{b})$  can be expressed in terms of a system of 4 vectors. This was shown in [11]. The main result is quoted next.

**THEOREM 3.1.** *Let two nonzero sequences  $\mathbf{a}, \mathbf{b} \in \mathbb{F}^N$  with  $\mathbf{a} \neq \pm \mathbf{b}$  be given. Then there exist a quadruple of nonnegative integers  $(d_j)_{j=1}^4$  satisfying  $\sum_{j=1}^4 d_j = 2(N+1)$ , vectors  $u_j \in \mathbb{F}^{d_j+1}$ , and signs  $\sigma_j \in \{1, -1\}$  ( $j = 1, 2, 3, 4$ ) satisfying  $\sum_{j=1}^4 \sigma_j = 0$  such that, for  $k = 1, \dots, N$  and  $\tau = \pm 1$ , the kernel of  $A_k^\tau(\mathbf{a}, \mathbf{b})$  consists of all vectors  $u$  for which  $u(t)$  is of the form*

$$(3.1) \quad u(t) = \sum_{j=1}^4 \xi_j(t) u_j(t),$$

where  $\xi_j$  is any vector from  $\mathbb{F}_{\tau \sigma_j}^{k-d_j}$  if  $k > d_j$  and  $\xi_j(t) = 0$  otherwise. The pairs  $(d_j, \sigma_j)$  are uniquely defined by  $(\mathbf{a}, \mathbf{b})$ , up to permutation.

The theorem generalizes a similar result for Toeplitz matrices (see [16]).

In particular,  $u_j$  belongs to the kernel of  $A_{d_j+1}(\mathbf{a}, \mathbf{b})$  if  $\sigma_j = 1$ , and it belongs to the kernel of  $A_{d_j+1}^-(\mathbf{a}, \mathbf{b})$  if  $\sigma_j = -1$ . The condition  $\sum_{j=1}^4 \sigma_j = 0$  means that two of the numbers  $\sigma_j$  are equal to 1 and two are equal to -1.

The system of vectors  $(u_j)_{j=1}^4$  or of the corresponding polynomials  $(u_j(t))_{j=1}^4$  is called *fundamental system* for the pair of vectors  $(\mathbf{a}, \mathbf{b})$ . The integer  $d_j$  is called *characteristic degree* of  $u_j$ , and  $\sigma_j$  *signature* of  $u_j$ .

Note that a T+H-matrix  $A_k(\mathbf{a}, \mathbf{b})$  might correspond to different sets of characteristic degrees depending on the representation (1.1). Recall that this representation is not unique due to the fact that the classes of Toeplitz and Hankel matrices have a nontrivial, 2-dimensional intersection.

From the theorem we conclude the following (see also [11]).

COROLLARY 3.2. Let  $A_n(\mathbf{a}, \mathbf{b})$  be an  $n \times n$  T+H matrix. Then the following statements are equivalent:

- (1)  $A_n(\mathbf{a}, \mathbf{b})$  is nonsingular.
- (2) The characteristic degrees  $d_j$  of  $A_n(\mathbf{a}, \mathbf{b})$  satisfy  $n - 1 \leq d_j \leq n + 1$  if  $\sigma_j = -1$  and  $n \leq d_j \leq n + 1$  if  $\sigma_j = 1$ .

We present two simple examples illustrating different situations.

EXAMPLE 1. Suppose that  $\mathbf{a} = (1, 1, 1)$  and  $\mathbf{b} = (0, 1, 0)$ . Then we have

$$A_2(\mathbf{a}, \mathbf{b}) = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}, \quad A_2^-(\mathbf{a}, \mathbf{b}) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Both matrices are nonsingular. Hence a fundamental system is provided by bases of the kernel of the matrices

$$A_3(\mathbf{a}, \mathbf{b}) = [1 \ 2 \ 1], \quad A_3^-(\mathbf{a}, \mathbf{b}) = [1 \ 0 \ 1].$$

We obtain a fundamental system

$$u_1 = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}, \quad u_2 = \begin{bmatrix} 2 \\ -1 \\ 0 \end{bmatrix}, \quad u_3 = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}, \quad u_4 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}.$$

The characteristic degrees are all equal to 2 and  $\sigma_1 = \sigma_2 = 1$ ,  $\sigma_3 = \sigma_4 = -1$ .

EXAMPLE 2. Suppose now that  $\mathbf{a} = (1, 1, 1)$  and  $\mathbf{b} = (0, 2, 0)$ . Then we have

$$A_2(\mathbf{a}, \mathbf{b}) = \begin{bmatrix} 1 & 3 \\ 3 & 1 \end{bmatrix}, \quad A_2^-(\mathbf{a}, \mathbf{b}) = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}.$$

The first matrix is nonsingular, the second singular. The vector  $u_1 = [1 \ 1]^T$  spanning the kernel of  $A_2^-(\mathbf{a}, \mathbf{b})$  provides the first vector in a fundamental system. The vectors  $u_2$  and  $u_3$  are vectors in the kernels of

$$A_3(\mathbf{a}, \mathbf{b}) = [1 \ 3 \ 1], \quad A_3^-(\mathbf{a}, \mathbf{b}) = [1 \ -1 \ 1].$$

respectively, that are linearly independent of the coefficient vectors of  $(t - 1)u_1(t)$  and  $(t + 1)u_2(t)$ , respectively. Finally,  $u_4$  is any vector of length 4 that is linearly independent of the coefficient vectors of  $(t^2 - 1)u_1(t)$ ,  $(t + 1)u_2(t)$ , and  $(t - 1)u_3(t)$ . One possibility is

$$u_2 = \begin{bmatrix} 3 \\ -1 \\ 0 \end{bmatrix}, \quad u_3 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad u_4 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

In particular, the characteristic degrees are  $d_1 = 1$ ,  $d_2 = d_3 = 2$ ,  $d_4 = 3$  and  $\sigma_2 = \sigma_4 = 1$ ,  $\sigma_1 = \sigma_3 = -1$ .

Clearly, the fundamental system for  $(\mathbf{a}, \mathbf{b})$  is not unique. The following proposition, which can easily be checked, describes how from one fundamental system another fundamental systems can be constructed. The freedom in choosing a fundamental system is important for the design of the algorithm in Section 4.

PROPOSITION 3.3. Suppose that  $(u_j(t))_{j=1}^4$  is a fundamental system for  $(\mathbf{a}, \mathbf{b})$ .  $(d_j)_{j=1}^4$  are the characteristic degrees and  $(\sigma_j)_{j=1}^4$  the corresponding signatures. Let one of the following conditions be satisfied:

- (1)  $d_p < d_j$
- (2)  $d_p = d_j$  and  $\sigma_p = \sigma_j$ .

Then a new fundamental system is obtained if  $u_j(t)$  is replaced by  $u_j(t) - \xi(t)u_p(t)$ , where  $\xi \in \mathbb{F}_{\sigma_p \sigma_j}^{d_j - d_p}$ .

It follows from Proposition 2.1 that if  $A_n(\mathbf{a}, \mathbf{b})$  is nonsingular, then  $A_{n+2}(\mathbf{a}, \mathbf{b})$  has kernel dimension 4 (note that the converse is not true). Any basis of this kernel is a generating system for  $A_n(\mathbf{a}, \mathbf{b})^{-1}$ . This can be obtained in the following way from a fundamental system (see [11]).

**PROPOSITION 3.4.** *Let  $A_n(\mathbf{a}, \mathbf{b})$  be nonsingular. Then a generating system  $(x_j)_{j=1}^4$  can be obtained from a fundamental system  $(u_j)_{j=1}^4$  according to*

$$x_j(t) = \begin{cases} (t + \sigma_j)u_j(t) & : d_j = n \\ (t^2 - 1)u_j(t) & : d_j = n - 1, \sigma_j = -1 \\ u_j(t) & : d_j = n + 1. \end{cases}$$

Let us briefly discuss the case of a field  $\mathbb{F}$  with characteristic 2. Looking at the proof Theorem 3.1 in [11] one can see that it can be extended to this case with the difference that  $\sum_{j=1}^4 d_j = 2(N+2)$  and that, of course, there are no signatures. The matrix  $A_n(\mathbf{a}, \mathbf{b})$  is nonsingular if and only if  $d_1 = d_2 = n$  and  $d_3 = d_4 = n+1$  and a generating system is given by  $x_j(t) = (1+t)u_j(t)$  for  $j = 1, 2$  and  $x_j(t) = u_j(t)$  for  $j = 3, 4$ .

#### 4. Fundamental systems after row extension

Let  $(u_j)_{j=1}^4$  be a fundamental system for  $(\mathbf{a}, \mathbf{b})$  ( $\mathbf{a} = (a_i)_{i=1}^m, \mathbf{b} = (b_i)_{i=1}^m \in \mathbb{F}^m$ ), let  $(d_j)_{j=1}^4$  be the corresponding characteristic degrees and  $(\sigma_j)_{j=1}^4$  the corresponding signatures. In this section we study how the fundamental system and the characteristic degrees change after extending  $\mathbf{a}$  and  $\mathbf{b}$ .

We consider extensions of  $\mathbf{a}$  and  $\mathbf{b}$  to  $(\mathbf{a}, \alpha)$  and  $(\mathbf{b}, \beta)$  for  $\alpha, \beta \in \mathbb{F}$ . These extensions correspond to one-row extensions of the matrices  $A_k(\mathbf{a}, \mathbf{b})$  at the bottom. We introduce the row vectors  $f_k^\sigma(\alpha, \beta)$  of length  $k$  by

$$f_k^\sigma(\alpha, \beta) = [a_{m+2-k} \dots a_m \ \alpha] + \sigma [\beta \ b_m \dots b_{m+2-k}],$$

where  $\sigma = \pm 1$ . Then the following is easily checked.

**PROPOSITION 4.1.** *Let  $\mathbf{a}' = (\mathbf{a}, \alpha)$ ,  $\mathbf{b}' = (\mathbf{b}, \beta)$  and  $u \in \ker A_k^\sigma(\mathbf{a}, \mathbf{b})$ . Then  $u$  belongs also to  $\ker A_k^\sigma(\mathbf{a}', \mathbf{b}')$  if and only if  $f_k^\sigma(\alpha, \beta)u = 0$ .*

The following proposition is crucial for the construction of the algorithms.

**PROPOSITION 4.2.** *Let  $u \in \ker A_k^\sigma(\mathbf{a}, \mathbf{b})$ ,  $\xi = (\xi_i)_{i=1}^{l-k+1} \in \mathbb{F}_\tau^{l-k+1}$ ,  $\tau = \pm 1$ , and  $v \in \mathbb{F}^l$  defined by  $v(t) = \xi(t)u(t)$ . Then*

$$f_l^{\sigma\tau}(\alpha, \beta)v = \tau\xi_1 f_k^\sigma(\alpha, \beta)u$$

**PROOF.** It is easily checked that for  $\xi(t) = 1 + t^{l-k}$  one has  $f_l^\sigma(\alpha, \beta)v = f_k^\sigma(\alpha, \beta)u$ , and for  $\xi(t) = 1 - t^{l-k}$  one has  $f_l^\sigma(\alpha, \beta)v = -f_k^{-\sigma}(\alpha, \beta)u$ . Furthermore, for  $\xi(t) = t^i + \sigma t^{l-k-i}$  with  $0 < i < m/2$  one has  $f_l^\sigma(\alpha, \beta)v = 0$ . The rest follows from the fact that the vectors corresponding to the polynomials  $t^i + \sigma t^{l-k-i}$  form a basis of  $\mathbb{F}_+^{l-k}$  or  $\mathbb{F}_-^{l-k}$ , respectively.  $\square$

We introduce the residuals  $r_j$  ( $j = 1, 2, 3, 4$ ) by

$$r_j = f_{d_j+1}^{\sigma_j}(\alpha, \beta)u_j \quad (j = 1, 2, 3, 4).$$

Clearly, one of the  $r_j$  must be different from 0, because otherwise all characteristic degrees would be unchanged after the extension which is not possible, according to Theorem 3.1.

We call a fundamental system  $(u_j)_{j=1}^4$   $(\alpha, \beta)$ -extension matched if one of the following is true:

**Case A.** Exactly one of the  $r_j$  is different from 0.

**Case B.** Exactly two of the  $r_j$  are different from 0, the corresponding characteristic degrees are equal, and the corresponding signatures are different.

We show how any fundamental system can be transformed into an extension matched one. Suppose  $r_p \neq 0$  and  $r_j \neq 0$ . If  $d_p < d_j$ , then we replace

$$(4.1) \quad u_j(t) \rightarrow r_p u_j(t) - r_j(1 + \sigma_p \sigma_j t^{j-p}) u_p(t).$$

By Proposition 3.3, this gives again a fundamental system, and by Proposition 4.2 the corresponding  $r_j$  is equal to 0. If  $d_p = d_j$  and  $\sigma_p = \sigma_j$ , then we replace

$$(4.2) \quad u_j(t) \rightarrow r_p u_j(t) - r_j u_p(t).$$

We have again a fundamental system and the new  $r_j$  is equal to 0. We conclude the following.

**PROPOSITION 4.3.** *For any given  $\alpha$  and  $\beta$ , a fundamental system can be transformed into an  $(\alpha, \beta)$ -extension matched one using transformations of the form (4.1) and (4.2).*

The main result of this section is the following. It can easily be checked using Propositions 4.1 and 4.2.

**THEOREM 4.4.** *Let  $(u_j)_{j=1}^4$  be an  $(\alpha, \beta)$ -extension matched fundamental system for  $(\mathbf{a}, \mathbf{b})$ ,  $(d_j)_{j=1}^4$  the corresponding characteristic degrees, and  $(\sigma_j)_{j=1}^4$  the corresponding signatures. Then a fundamental system  $(u'_j)_{j=1}^4$ , the characteristic degrees  $(d'_j)_{j=1}^4$ , and the signatures  $(\sigma'_j)_{j=1}^4$  for  $((\mathbf{a}, \alpha), (\mathbf{b}, \beta))$  are given as follows:*

Case (A): Let  $r_p \neq 0$  and  $r_j = 0$  for  $j \neq p$ . Then  $u'_j(t) = u_j(t)$  and  $d'_j = d_j$  for  $j \neq p$ , and  $u'_p(t) = tu_p(t)$  and  $d'_p = d_p + 2$ .

Case (B): Let  $r_p r_q \neq 0$ ,  $d_p = d_q$ ,  $\sigma_p = 1$ ,  $\sigma_q = -1$ , and  $r_j = 0$  for  $j \neq p, q$ . Then  $u'_j(t) = u_j(t)$  and  $d'_j = d_j$  for  $j \neq p, q$ , and

$$\begin{aligned} u'_p(t) &= (1+t)r_q u_p(t) - (1-t)r_p u_q(t) \\ u'_q(t) &= (1-t)r_q u_p(t) - (1+t)r_p u_q(t). \end{aligned}$$

$d'_p = d'_q = d_p + 1$ . In all cases  $\sigma'_j = \sigma_j$  ( $j = 1, 2, 3, 4$ ).

Similarly extensions  $\mathbf{a} \rightarrow (\alpha, \mathbf{a})$  and  $\mathbf{b} \rightarrow (\beta, \mathbf{b})$ , which correspond to one-row extensions at the top of  $A_k(\mathbf{a}, \mathbf{b})$ , can be treated. The only difference is that instead of  $r_j$  one has to consider the residuals

$$s_j = ([\alpha \ a_1 \ \dots \ a_{d_j}] + \sigma_j [b_{d_j} \ \dots \ b_1 \ \beta]) u_j$$

for  $j = 1, 2, 3, 4$ .

It is important to observe that a pair of the extensions just considered, i.e. an extension by a row at the top and a row at the bottom is equivalent to an extension by a column at the left and a column at the right. With the help of these kinds of extensions one can obtain a so-called WZ-factorization of the inverse T+H matrix, provided that all central submatrices of the T+H-matrix are nonsingular. A WZ-factorization is similar to a UL-factorization, but instead of the upper triangular

factor one has an bow-tie matrix and instead of the lower triangular factor one has a hourglass matrix<sup>2</sup> This is discussed in the recent paper [23].

If one wants to construct a triangular factorization of  $A_n(\mathbf{a}, \mathbf{b})^{-1}$  one has also to consider single column extensions, which correspond to extensions of  $\mathbf{a}$  and  $\mathbf{b}$  to opposite sides, i.e. for example extensions  $\mathbf{a} \rightarrow (\mathbf{a}, \alpha)$  and  $\mathbf{b} \rightarrow (\beta, \mathbf{b})$ . This seems to be more difficult to describe. One possibility is to do it via one column deletion plus two row extensions, one at the top and one at the bottom. Therefore, we study also change of a fundamental system after column deletion is in the next section.

If the characteristic of the field  $\mathbb{F}$  is equal to 2 only Case A is possible. Otherwise the arguments of this section transfer to this case.

## 5. Fundamental systems after column deletion

We study the change of a fundamental system after deleting one column at the right of the T+H-Matrices  $A_k(\mathbf{a}, \mathbf{b})$ . Suppose that, as before,  $\mathbf{a} = (a_i)_{i=1}^m$  and  $\mathbf{b} = (b_i)_{i=1}^m$ . A column deletion at the right corresponds to a reduction of the sequences  $\mathbf{a}$  and  $\mathbf{b}$  to  $\mathbf{a}' = (a_i)_{i=1}^{m-1}$  and  $\mathbf{b}' = (b_i)_{i=2}^m$ .

Let  $z(x)$  denote the last component of the vector  $x \in \mathbb{F}^k$ . If  $x \in \ker A_k(\mathbf{a}, \mathbf{b})$  and  $z(x) = 0$ , then the vector obtained after deleting the last component of  $x$  (which is zero) belongs to  $\ker A_{k-1}(\mathbf{a}', \mathbf{b}')$  and, vice versa, any vector from  $\ker A_{k-1}(\mathbf{a}', \mathbf{b}')$  is of this form.

If  $(u_j)_{j=1}^4$  is a fundamental system for  $(\mathbf{a}, \mathbf{b})$ , then we denote  $z_j = z(u_j)$ . The fundamental system will be called *reduction matched* if one of the following is true:

**Case A.** Exactly one of the  $z_j$  is different from 0.

**Case B.** Exactly two of the  $z_j$  are different from 0, the corresponding characteristic degrees are equal, and the corresponding signatures are different.

Similarly as for extension matching, any fundamental system can be transformed into a reduction matched one via transformations

$$u_j(t) \rightarrow z_p u_j(t) - z_j(1 + \sigma_p \sigma_j t^{j-p}) u_p(t).$$

if  $d_p < d_j$  and

$$u_j(t) \rightarrow z_p u_j(t) - z_j u_p(t)$$

if  $d_p = d_j$  and  $\sigma_p = \sigma_j$ .

**THEOREM 5.1.** Let  $(u_j)_{j=1}^4$  be a reduction matched fundamental system for  $(\mathbf{a}, \mathbf{b})$ ,  $(d_j)_{j=1}^4$  the corresponding characteristic degrees, and  $(\sigma_j)_{j=1}^4$  the corresponding signatures. Then a fundamental system  $(u'_j)_{j=1}^4$ , the characteristic degrees  $(d'_j)_{j=1}^4$  and the signatures  $(\sigma'_j)_{j=1}^4$  for  $(\mathbf{a}', \mathbf{b}')$  are given as follows:

Case (A): Let  $z_p \neq 0$  and  $z_j \neq 0$  for  $j \neq p$ . Then  $u'_j(t) = u_j(t)$  and  $d'_j = d_j - 1$  for  $j \neq p$ , and  $u'_p(t) = tu_p(t)$  and  $d'_p = d_p + 1$ .

Case (B): Let  $z_p z_q \neq 0$ ,  $d_p = d_q$ ,  $\sigma_p = 1$ ,  $\sigma_q = -1$ , and  $z_j = 0$  for  $j \neq p, q$ . Then  $u'_j(t) = u_j(t)$  and  $d'_j = d_j - 1$  for  $j \neq p, q$ , and

$$\begin{aligned} u'_p(t) &= (1+t)z_q u_p(t) - (1-t)z_p u_q(t) \\ u'_q(t) &= (1-t)z_q u_p(t) - (1+t)z_p u_q(t), \end{aligned}$$

$d'_p = d'_q = d_p$ . In all cases  $\sigma'_j = \sigma_j$  ( $j = 1, 2, 3, 4$ ).

---

<sup>2</sup>Concerning WZ-factorization of tridiagonal matrices see [3], of general matrices see [26], of symmetric Toeplitz matrices see [2], and skewsymmetric Toeplitz matrices see [22].

If the characteristic of the field  $\mathbb{F}$  is equal to 2 only Case A is possible. Otherwise the arguments of this section transfer to this case.

## 6. Levinson-type algorithm

We describe now how the fundamental system of a pair  $(\mathbf{a}, \mathbf{b})$ ,  $\mathbf{a} = (a_i)_{i=1}^N$  and  $\mathbf{b} = (b_i)_{i=1}^N$  can be constructed. We assume that  $\mathbf{a} \neq \pm \mathbf{b}$ . This is no restriction of generality, since the cases  $\mathbf{a} = \mathbf{b}$  and  $\mathbf{a} = -\mathbf{b}$  reduce to pure Toeplitz matrices. The pure Toeplitz case was already considered in [6] (see also [16]).

For initialization of the algorithm we choose a  $p$  such that  $a_p \neq \pm b_p$ . In view of our assumption such a  $p$  exists. We start the recursion with  $\mathbf{a}^{(1)} = (a_p)$ ,  $\mathbf{b}^{(1)} = (b_p)$ . A fundamental system of  $(\mathbf{a}^{(1)}, \mathbf{b}^{(1)})$  is given by  $u_1^{(1)}(t) = u_3^{(1)}(t) = 1$ ,  $u_2^{(1)}(t) = u_4^{(1)}(t) = t$ , or in vector form

$$u_1^{(1)} = u_3^{(1)} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad u_2^{(1)} = u_4^{(1)} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

The characteristic degrees are all equal to 1, and  $\sigma_1 = \sigma_2 = 1$ ,  $\sigma_3 = \sigma_4 = -1$ .

Now we assume that a fundamental system  $u_j^{(m)}$  for  $(\mathbf{a}^{(m)}, \mathbf{b}^{(m)})$ , where  $\mathbf{a}^{(m)} = (a_i)_{i=l+1}^{l'}, \mathbf{b}^{(m)} = (b_i)_{i=l+1}^{l'}$ , and  $m = l' - l$ , is given. Let  $d_j^{(m)}$  denote the corresponding characteristic degrees and  $\sigma_j$  the signatures. Note that we do not need a superscript  $(m)$  for  $\sigma_j$  since the signatures do not change during the algorithm.

For an extension of  $\mathbf{a}$  and  $\mathbf{b}$  to the right one has to carry out the following steps:

1. Compute the residuals

$$r_j^{(m)} = \left( [a_{l'+1-d_j^{(m)}} \dots a_{l'} \ a_{l'+1}] + \sigma_j [b_{l'+1} \ b_{l'} \dots b_{l'+1-d_j^{(m)}}] \right) u_j^{(m)}$$

for  $j = 1, 2, 3, 4$ . If the previous step was Case A, then only 3 new residuals have to be computed. The nonzero one from the previous step is still the same.

2. Transform the fundamental system into an  $(a_{l'+1}, b_{l'+1})$ -extension matched one by replacing  $u_j^{(m)}$  by

$$r_p^{(m)} u_j^{(m)} - r_j^{(m)} \left( \begin{bmatrix} u_p^{(m)} \\ \mathbf{0} \end{bmatrix} + \sigma_p \sigma_j \begin{bmatrix} \mathbf{0} \\ u_p^{(m)} \end{bmatrix} \right)$$

if  $r_p^{(m)} r_j^{(m)} \neq 0$  and  $d_p^{(m)} < d_j^{(m)}$ , and replacing  $u_j^{(m)}$  by

$$r_p^{(m)} u_j^{(m)} - r_j^{(m)} u_p^{(m)}$$

if  $d_p^{(m)} = d_j^{(m)}$  and  $\sigma_p^{(m)} = \sigma_j^{(m)}$ . The residuals of the new vectors vanish. Now it can be seen whether one has Case A (one residual is different from zero) or Case B (two residuals are different from zero, the corresponding characteristic degrees coincide and the signatures are opposite).

3. Apply the formulas of Theorem 5.1 to obtain a fundamental system  $(u_j^{(m+1)})_{j=1}^4$  for  $(\mathbf{a}^{(m+1)}, \mathbf{b}^{(m+1)})$ , where  $\mathbf{a}^{(m+1)} = (a_i)_{i=l+1}^{l'+1}$  and  $\mathbf{b}^{(m+1)} = (b_i)_{i=l+1}^{l'+1}$ . In vector form these formulas can be written in Case A, if  $r_p^{(m)} \neq 0$ , as

$$u_j^{(m+1)} = u_j^{(m)} \quad (j \neq p), \quad u_p^{(m+1)} = \begin{bmatrix} 0 \\ u_p^{(m)} \\ 0 \end{bmatrix}$$

and in Case B, if  $r_p^{(m)} r_q^{(m)} \neq 0$ ,  $u_j^{(m+1)} = u_j^{(m)}$  for  $j \neq p, q$ ,

$$u_p^{(m+1)} = r_q^{(m)} \left( \begin{bmatrix} u_p^{(m)} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ u_p^{(m)} \end{bmatrix} \right) - r_p^{(m)} \left( \begin{bmatrix} u_q^{(m)} \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ u_q^{(m)} \end{bmatrix} \right)$$

and

$$u_q^{(m+1)} = r_q^{(m)} \left( \begin{bmatrix} u_p^{(m)} \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ u_p^{(m)} \end{bmatrix} \right) - r_p^{(m)} \left( \begin{bmatrix} u_q^{(m)} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ u_q^{(m)} \end{bmatrix} \right).$$

For a left extension to  $(\mathbf{a}^{(m)}, \mathbf{b}^{(m)})$  one has to compute

$$s_j^{(m)} = \left( [a_l \ a_{l+1} \ \dots \ a_{l+d_j^{(m)}}] + \sigma_j [b_{l+d_j^{(m)}} \ \dots \ b_{l+1} \ b_l] \right) u_j^{(m)}$$

instead. Then the system  $(u_j^{(m)})_{j=1}^4$  has to be transformed into a  $(a_l, b_l)$ -extension matched one and the “left” analog of Theorem 5.1 has to be applied to obtain a fundamental system for  $(\mathbf{a}^{(m+1)}, \mathbf{b}^{(m+1)})$ , where  $\mathbf{a}^{(m+1)} = (a_i)_{i=l}^{l'}$  and  $\mathbf{b}^{(m+1)} = (b_i)_{i=l}^{l'}$ . After  $N - 1$  steps of the algorithm one arrives at a fundamental system for  $(\mathbf{a}, \mathbf{b})$ .

If one is interested to have fundamental systems of vectors corresponding to submatrices of the original matrix during the algorithm one has to start in the middle of the sequences  $\mathbf{a}$  and  $\mathbf{b}$  and has to switch between right and left steps.

Let us point out that the algorithm just proposed is division free. This is convenient if the entries of the original matrix are integers, so integer arithmetics can be used. If  $\mathbb{F}$  are the real or complex numbers, then, of course, the magnitude of the vectors may blow up in magnitude. To avoid this the algorithm can be modified by including a scaling operation. This could be normalizing of  $u_j^{(m)}$  to norm 1 or dividing by the last nonzero coefficient of  $u_j^{(m)}$ .

In case the  $\mathbb{F}$  is the field of real or complex numbers in the generic case all submatrices have full rank. In this case all residuals are nonzero, the 4 characteristic degrees differ at most by 1, and one has Case B in each step.

In this case the complexity of the algorithm can easily be estimated. Let us consider first the case when all characteristic degrees are equal to  $d$ . For the next fundamental system one has to compute 4 new residuals, which amounts in 4 inner products of length  $m$ , i.e.  $4m$  additions and  $4m$  multiplications. Then one has to compute 2 linear combinations of vectors for the matching operation, which results in  $2m$  additions and  $4m$  multiplications. Finally, one has to compute 4 vector sums and 2 linear combinations, which requires  $6m$  additions and  $4m$  multiplications. In the next step, when 2 characteristic degrees are equal to  $d$  and 2 are equal to  $d+1$ , we need 2 more vector additions for the matching operation. Summing up, the overall complexity will be about  $6.5N^2$  additions and  $6N^2$  multiplications. Note that the algorithm can slightly be modified by also allowing divisions. This reduces the number of multiplications from  $6N^2$  to  $4N^2$ .

In the general case the complexity depends on the rank profile of the submatrices, but it becomes clear from the formulas that the complexity will not be larger than in the generic case.

Once a fundamental system for  $(\mathbf{a}, \mathbf{b})$  with  $\mathbf{a}, \mathbf{b} \in \mathbb{F}^{2n-1}$  is computed, the characteristic degree indicate whether the matrix  $A_n(\mathbf{a}, \mathbf{b})$  is nonsingular or not and, in case that it is nonsingular, a generating system for  $A_n(\mathbf{a}, \mathbf{b})^{-1}$  is given by Proposition 3.4. The next step would be to transform the generating system into

a canonical one. This is the only place in the procedure where divisions have to be carried out to find the inverse of a  $4 \times 4$  matrix. If the same is done for the pair  $(\mathbf{a}, J_{2n-1}\mathbf{b})$  all data included in the inversion formula are evaluated.

If the characteristic of  $\mathbb{F}$  is equal to 2, then the algorithm simplifies. However, the initialization should be different. One can start with a nonsingular  $2 \times 2$  matrix, i.e. with  $m = 3$ . In this case  $d_1^{(3)} = d_2^{(3)} = 2$  and  $d_3^{(3)} = d_4^{(3)} = 3$ .

## 7. Schur-type algorithm

The recursions of the fundamental systems described in the previous section cannot be completely parallelized, since they include inner product calculations to compute the residuals  $r_j^{(m)}$  and  $s_j^{(m)}$ . This is one of several motivations to consider Schur-type algorithms, which precompute the residuals and lead to algorithms with  $O(n)$  parallel complexity. Other motivations to consider Schur-type algorithms are that they provide a factorization of the original matrix rather than its inverse and they are, as a rule, more stable for ill-conditioned matrices. In this section we present a Schur-type algorithm that can be immediately derived from the recursions in Theorem 5.1.

Suppose that  $\mathbf{a} = (a_i)_{i=1}^N$ ,  $\mathbf{b} = (b_i)_{i=1}^N$ ,  $\mathbf{a}^{(m)} = (a_i)_{i=l+1}^{l'}$ ,  $\mathbf{b}^{(m)} = (b_i)_{i=l+1}^{l'}$  with  $l' - l = m$ . Let  $g_{ik}^\sigma$  denote the  $i$  th row of the matrix  $A_k^\sigma(\mathbf{a}, \mathbf{b})$ , i.e.

$$g_{ik}^\sigma = [a_i \dots a_{i+k-1}] + \sigma [b_{i+k-1} \dots b_i].$$

For any  $u_j^{(m)}$ , we introduce the full set of  $N - m$  residuals

$$r_{ij}^{(m)} = g_{l'-d_j^{(m)}+i,d_j^{(m)}+1}^{\sigma_j} u_j^{(m)} \quad (i = 1, \dots, N - l')$$

and

$$s_{ij}^{(m)} = g_{l+1-i,d_j^{(m)}+1}^{\sigma_j} u_j^{(m)} \quad (i = 1, \dots, l)$$

where  $j = 1, 2, 3, 4$ . In particular, we have  $r_{1j}^{(m)} = r_j^{(m)}$  and  $s_{1j}^{(m)} = s_j^{(m)}$ .

The recursions for the residuals can be carried out independently from those for the fundamental systems. Therefore, it is reasonable to speak about an *extension matched* or *recursion matched residual system*, which is defined in the same way as extension and recursion matched fundamental systems.

The matching operations (4.1) and (4.2) correspond to the following operations on the residuals:

$$r_{ij}^{(m)} \rightarrow r_p^{(m)} r_{ij}^{(m)} - r_j^{(m)} (r_{i-j+p}^{(m)} + \sigma_p \sigma_j r_{ip}^{(m)})$$

and

$$r_{ij}^{(m)} \rightarrow r_p^{(m)} r_{ij}^{(m)} - r_j^{(m)} r_{ip}^{(m)}.$$

As an immediate consequence of Theorem 4.4 we obtain the following.

**THEOREM 7.1.** *Let  $((r_{ij}^{(m)})_{i=1}^{N-l'})_{j=1}^4$  be an  $(a_{l'+1}, b_{l'+1})$ -extension matched residual system for  $(\mathbf{a}^{(m)}, \mathbf{b}^{(m)})$ . Then a residual system  $((r_{ij}^{(m+1)})_{i=1}^{N-l'-1})_{j=1}^4$  for  $(\mathbf{a}^{(m+1)}, \mathbf{b}^{(m+1)})$  is given as follows:*

*Case (A): Let  $r_p^{(m)} \neq 0$  and  $r_j^{(m)} = 0$  for  $j \neq p$ . Then*

$$r_{ij}^{(m+1)} = r_{i+1,j}^{(m)}, \quad s_{ij}^{(m+1)} = s_{ij}^{(m)}$$

*for  $j \neq p$ , and*

$$r_{ip}^{(m+1)} = r_{ip}^{(m)}, \quad s_{ip}^{(m+1)} = s_{i+1,p}^{(m)}.$$

Case (B): Let  $r_p^{(m)} r_q^{(m)} \neq 0$ ,  $d_p^{(m)} = d_q^{(m)}$ ,  $\sigma_p = 1$ ,  $\sigma_q = -1$ , and  $r_j^{(m)} = 0$  for  $j \neq p, q$ . Then

$$r_{ij}^{(m+1)} = r_{i+1,j}^{(m)}, \quad s_{ij}^{(m+1)} = s_{ij}^{(m)}$$

for  $j \neq p, q$ , and

$$\begin{aligned} r_{ip}^{(m+1)} &= r_q^{(m)}(r_{ip}^{(m)} + r_{i+1,p}^{(m)}) + r_p^{(m)}(r_{iq}^{(m)} - r_{i+1,q}^{(m)}) \\ r_{iq}^{(m+1)} &= r_q^{(m)}(r_{ip}^{(m)} - r_{i+1,p}^{(m)}) - r_p^{(m)}(r_{iq}^{(m)} + r_{i+1,q}^{(m)}) \\ s_{ip}^{(m+1)} &= r_q^{(m)}(s_{ip}^{(m)} + s_{i-1,p}^{(m)}) + r_p^{(m)}(s_{iq}^{(m)} - s_{i-1,q}^{(m)}) \\ s_{iq}^{(m+1)} &= r_q^{(m)}(s_{ip}^{(m)} - s_{i-1,p}^{(m)}) - r_p^{(m)}(s_{iq}^{(m)} + s_{i-1,q}^{(m)}). \end{aligned}$$

The characteristic degrees change as described in Theorem 5.1 and the signatures remain unchanged. Similar theorems hold for left extensions of  $(\mathbf{a}^{(m)}, \mathbf{b}^{(m)})$  and for single column deletions.

Starting the recursion with  $\mathbf{a}^{(1)} = (a_p)$  and  $\mathbf{b}^{(1)} = (b_p)$ ,  $b_p \neq \pm a_p$ , means that we start with the residuals

$$\begin{aligned} r_{i1}^{(1)} &= a_{p-1+i} + b_{p+i}, & r_{i2}^{(1)} &= a_{p+i} + b_{p-1+i}, \\ r_{i3}^{(1)} &= a_{p-1+i} - b_{p+i}, & r_{i4}^{(1)} &= a_{p+i} - b_{p-1+i} \end{aligned}$$

and

$$\begin{aligned} s_{i1}^{(1)} &= a_{p-i} + b_{p+1-i}, & s_{i2}^{(1)} &= a_{p+1-i} + b_{p-i}, \\ s_{i3}^{(1)} &= a_{p-i} - b_{p+1+i}, & s_{i4}^{(1)} &= a_{p+1-i} - b_{p-i}. \end{aligned}$$

The recursion in Theorem 7.1 can be used in place of the first step of the Levinson algorithm in Section 6. The complexity of the resulting algorithm will be slightly higher, but after this replacement all calculations can be done in parallel, so that the overall parallel complexity will be  $O(N)$ .

If all central submatrices of a T+H-matrix  $A$  are nonsingular, then the residuals provide the factors of a ZW-factorization of  $A$  (see [23]). This factorization can also be used to solve linear systems without computing a fundamental system. Therefore, it is desirable to have some kind of “generalized ZW-factorization” of a general T+H-matrix. It will be shown in a forthcoming paper that for the case of a centro-symmetric T+H-matrix a natural generalization does exist. For the general case this question is still open.

The combination of the recursions in Theorem 5.1 and 7.1 can be used to design an algorithm with sequential complexity less than  $O(N^2)$ , provided that a fast polynomial multiplication in  $\mathbb{F}$  is available. Let us briefly explain this.

We introduce the row vector polynomial

$$u^{(m)}(t) = [ u_1^{(m)}(t) \ u_2^{(m)}(t) \ u_3^{(m)}(t) \ u_4^{(m)}(t) ]$$

Then the recursion in Theorem 5.1 can be written in the form

$$u^{(m+1)}(t) = u^{(m)}(t)\Phi_m^{m+1}(t)$$

for some  $4 \times 4$  matrix polynomial  $\Phi_m^{m+1}(t)$ . For  $m' > m$  we have now

$$u^{(m')}(t) = u^{(m)}(t)\Phi_m^{m'}(t)$$

for some matrix polynomial  $\Phi_m^{m'}(t)$ . The corresponding residuals are related via

$$r^{(m')}(t) = r^{(m)}(t)\tilde{\Phi}_m^{m'}(t^{-1}),$$

where  $\tilde{\Phi}_m^{m'}(t)$  is directly related to  $\Phi_m^{m'}(t)$ . Since, for  $m < m' < m''$ ,

$$\Phi_m^{m''}(t) = \Phi_m^{m'}(t)\Phi_{m'}^{m''}(t)$$

a divide and conquer strategy can be applied. If now multiplication of polynomials of degree  $m$  can be carried out with complexity  $M(m)$ , then the overall complexity for the computation of the fundamental system will be  $O(M(N) \log N)$ . If  $\mathbb{F}$  is the field of real or complex numbers, then one has  $M(m) = m \log m$ , so that the overall complexity is  $O(N \log^2 N)$ . We refrain from presenting the details of this “superfast” algorithm, because it is unclear whether it is really practical.

## References

- [1] D. BINI, V. PAN, *Polynomial and matrix computations, Vol.1. Fundamental algorithms*. Birkhäuser, Boston, 1994.
- [2] C. J. DEMEURE, *Bowtie factors of Toeplitz matrices by means of split algorithms*. IEEE Trans. Acoustics Speech, and Signal Processing, ASSP- 37, 10 (1989), 1601–1603.
- [3] D. J. EVANS, M. HATZOPoulos, *A parallel linear systems solver*. Internat. J. Comput. Math., 7, 3 (1979), 227–238.
- [4] I. GOHBERG, T. KAILATH, V. OLSHEVSKY, *Fast Gaussian elimination with partial pivoting for matrices with displacement structure*. Math.of Comp. 64 (1995), 1557–1576.
- [5] I. GOHBERG, I. KOLTRACHT, *Efficient algorithms for Toeplitz-plus-Hankel matrices*. Integral Equ. and Operator Theory, 12, 1 (1989), 136–142.
- [6] G. HEINIG, *Inversion of Toeplitz and Hankel matrices with singular sections*. Wiss. Zeitschr. d. TH Karl-Marx-Stadt, 25, 3 (1983), 326–333.
- [7] G. HEINIG, *Partial indices for Toeplitz-like operators*. Integral Equ. and Operator Theory, 8 (1985), 805–824.
- [8] G. HEINIG, *Formulas and algorithms for block Hankel matrix inversion and partial realization*, In: *Progress in Systems and Control*, vol.5, Birkhäuser 1990. pp.79–90.
- [9] G. HEINIG, *Inversion of Toeplitz-like matrices via generalized Cauchy matrices and rational interpolation*, In: *Systems and Network: Mathematical Theory and Applications*. Akademie Verlag 1994, vol.2. 707–711.
- [10] G. HEINIG, *Generalized Cauchy-Vandermonde matrices*. Linear Algebra Appl., 270 (1997), 45–77.
- [11] G. HEINIG, *Kernel structure of Toeplitz-plus-Hankel matrices*. Linear Algebra Appl., 340 (2002), 1–13.
- [12] G. HEINIG, A. BOJANCZYK, *Transformation techniques for Toeplitz and Toeplitz-plus-Hankel matrices*, I. Linear Algebra Appl., 254 (1997), 193–226; II. Linear Algebra Appl. 27 (1998), 11–36.
- [13] G. HEINIG, P. JANKOWSKI, *Parallel and superfast algorithms for Hankel systems of equations*. Numerische Mathematik, 58 (1990), 109–127.
- [14] G. HEINIG, P. JANKOWSKI, K. ROST, *Fast inversion algorithms of Toeplitz-plus-Hankel matrices*. Numer. Math., 52 (1988), 665–682.
- [15] G. HEINIG, V. OLSHEVSKY, *The Schur algorithm for matrices with a Hessenberg displacement structure*, In: V. OLSHEVSKY (Ed.), *Structured Matrices in Mathematics, Computer Science, and Engineering*, vol.2, AMS-Series Contemporary Mathematics. 2001. pp.3–15.
- [16] G. HEINIG, K. ROST, *Algebraic Methods for Toeplitz-like Matrices and Operators*. Birkhäuser Verlag, Basel, Boston, Stuttgart. 1984.
- [17] G. HEINIG, K. ROST, *On the inverses of Toeplitz-plus-Hankel matrices*. Linear Algebra Appl., 106 (1988), 39–52.
- [18] G. HEINIG, K. ROST, *Matrix representation of Toeplitz-plus-Hankel matrix inverses*. Linear Algebra Appl., 113 (1989), 65–78.
- [19] G. HEINIG, K. ROST, *DFT representations of Toeplitz-plus-Hankel Bezoutians with application to fast matrix-vector multiplication*. Linear Algebra Appl., 284 (1998), 157–175.
- [20] G. HEINIG, K. ROST, *Hartley transform representations of inverses of real Toeplitz-plus-Hankel matrices*, Numer. Funct. Anal. and Optimization, 21 (2000), 175–189.

- [21] G. HEINIG, K. ROST, *Efficient inversion formulas for Toeplitz-plus-Hankel matrices using trigonometric transformations*, In: V. OLSHEVSKY (Ed.), *Structured Matrices in Mathematics, Computer Science, and Engineering*, vol. 2, AMS-Series *Contemporary Mathematics* 281 (2001), 247–264.
- [22] G. HEINIG, K. ROST, *Fast algorithms for skewsymmetric Toeplitz matrices*, Operator Theory: Advances and Applications, Birkhäuser Verlag, Basel, Boston, Berlin, vol. 135 (2002), 193–208.
- [23] G. HEINIG, K. ROST, *New fast algorithms for Toeplitz-plus-Hankel matrices*, submitted.
- [24] G.A. MERCHANT, T.W. PARKS, *Efficient solution of a Toeplitz-plus-Hankel coefficient matrix system of equations*, IEEE Transact. on Acoust.Speech Sign.Process., 30, 1 (1982), 40–44.
- [25] A.B. NERSESSIAN, A.A. PAPOYAN, *Construction of the matrix inverse to the sum of Toeplitz and Hankel matrices* (in Russian), Izv. AN Arm.SSR, 18, 2 (1983), 150–160.
- [26] S. CHANDRA SEKHARA RAO, *Existence and uniqueness of WZ factorization*, Parallel Comp., 23, 8 (1997), 1129–1139.
- [27] A.H. SAYED, H. LEV-ARI, T. KAILATH, *Fast triangular factorization of the sum of quasi-Toeplitz and quasi-Hankel matrices*, Linear Algebra Appl., 191 (1993), 77–106.
- [28] A.E. YAGLE, *New analogs of split algorithms for arbitrary Toeplitz-plus-Hankel matrices*, IEEE Trans. Signal Process., 39, 11 (1991), 2457–2463.
- [29] C.J. ZAROWSKI, *A Schur algorithm and linearly connected processor array for Toeplitz-plus-Hankel matrices*, IEEE Transact. on Inf. Theory, 40, 8 (1992), 2065–2078.

KUWAIT UNIVERSITY, DEPT. OF MATH. & COMP. SCI., P.O.Box 5969, SAFAT 13060, KUWAIT  
*E-mail address:* georg@mcs.sci.kuniv.edu.kw

*This page intentionally left blank*

## A Lanczos-type Algorithm for the QR Factorization of Cauchy-like Matrices

Dario Fasino and Luca Gemignani

**ABSTRACT.** We present a fast algorithm for computing the  $QR$  factorization of Cauchy-like matrices with real nodes. The algorithm is based on the existence of certain generalized recurrences among the columns of  $Q$  and  $R^T$ , does not require squaring the matrix, and fully exploits the displacement structure of Cauchy-like matrices. Moreover, we describe a class of Cauchy-like matrices admitting an explicit polar decomposition.

### 1. Introduction

Let  $\{x_i\}_{1 \leq i \leq n}$  and  $\{y_i\}_{1 \leq i \leq n}$  be two sets of real distinct points, where  $x_i \neq y_j$  for  $1 \leq i, j \leq n$ . Moreover, let  $\{v_i\}_{1 \leq i \leq n}$  and  $\{w_i\}_{1 \leq i \leq n}$  be arbitrary nonzero numbers. The matrix

$$(1.1) \quad C = \begin{pmatrix} \frac{v_1 w_1}{x_1 - y_1} & \frac{v_2 w_1}{x_1 - y_2} & \cdots & \frac{v_n w_1}{x_1 - y_n} \\ \frac{v_1 w_2}{x_2 - y_1} & \frac{v_2 w_2}{x_2 - y_2} & \cdots & \frac{v_n w_2}{x_2 - y_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{v_1 w_n}{x_n - y_1} & \frac{v_2 w_n}{x_n - y_2} & \cdots & \frac{v_n w_n}{x_n - y_n} \end{pmatrix}$$

belongs to the class of *Cauchy-like matrices* [8, 12, 17]. Matrices with the structure (1.1) are also called *matrices of Cauchy type* [2], and *generalized Cauchy matrices* [13, 14]. Such matrices appear as building blocks in the design of fast algorithms for displacement structured matrices [2, 8, 12, 13, 14, 17], but also in connection with rational interpolation and approximation problems [1, 21] as well as in the discretization of integral operators [15, 24].

This paper is concerned with the efficient computation of a  $QR$  factorization of the matrix  $C$ , where  $Q$  is orthogonal and  $R$  is upper triangular. An explicit application of such a factorization to solve a least square approximation problem

---

2000 *Mathematics Subject Classification*. Primary 15A23, 65F23.

*Key words and phrases*. Cauchy matrices, QR factorization, fast algorithms.

involving rational functions is shown below. Consider the discrete inner product

$$(1.2) \quad \langle \phi, \psi \rangle = \sum_{i=1}^n w_i^2 \phi(x_i) \psi(x_i),$$

and the associated norm  $\|\phi\| = \langle \phi, \phi \rangle^{1/2}$ . Let  $\phi_j(x) = (x - y_j)^{-1}$ , for  $j = 1, \dots, m$ , with  $m \leq n$ . Given a function  $f$  defined on the nodes  $x_i$ , the least square problem

$$(1.3) \quad \min_{a_1, \dots, a_m} \left\| \sum_{i=1}^m a_i \phi_i - f \right\|^2$$

is equivalent to the solution, in the least square sense, of the overdetermined system

$$\begin{pmatrix} \frac{w_1}{x_1 - y_1} & \dots & \frac{w_1}{x_1 - y_m} \\ \vdots & \vdots & \vdots \\ \frac{w_n}{x_n - y_1} & \dots & \frac{w_n}{x_n - y_m} \end{pmatrix} \begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix} = \begin{pmatrix} w_1 f(x_1) \\ \vdots \\ w_n f(x_n) \end{pmatrix},$$

whose coefficient matrix  $\hat{C}$  of order  $n \times m$  is built by the first  $m$  columns of the matrix  $C$  of (1.1), with  $v_i = 1$ . Problem (1.3) can therefore be solved by a standard algorithm exploiting the  $QR$  factorization of the matrix  $\hat{C}$  [10], and the latter can be obtained from that of  $C$  by simply deleting the last  $n - m$  columns from its triangular factor  $R$ . Moreover, it is worth realizing that the computation of  $C = QR$  provides a means to orthogonalize the set of functions  $\{\phi_1, \dots, \phi_m\}$  with respect to the inner product (1.2). In fact, if  $R^{-1} \equiv (\hat{r}_{i,j})$  then the functions

$$\Phi_j(x) = \sum_{i=1}^j \hat{r}_{i,j} \phi_i(x)$$

are a family of orthonormal rational functions uniquely defined (apart of the sign) by the condition  $\Phi_j \in \text{Span}\{\phi_1, \dots, \phi_j\}$ : furthermore, we have  $Q \equiv (w_i \Phi_j(x_i))$ .

Throughout this paper, we adopt the following notations: Let  $e_i$  be the  $i$ th canonical basis vector. Given a vector  $\mathbf{v} = [v_1, \dots, v_n]^T$  let  $D_v = \text{Diag}[v_1, \dots, v_n]$  be the  $n \times n$  diagonal matrix with diagonal entries  $v_i$ . Then, it is easily seen that

$$(1.4) \quad D_x C - C D_y = \mathbf{w} \mathbf{v}^T,$$

which provides a so-called *displacement equation* for the Cauchy-like matrix  $C$  [8, 16, 18]. Since  $C^T$  is also a Cauchy-like matrix,

$$(1.5) \quad D_{-y} C^T - C^T D_{-x} = \mathbf{v} \mathbf{w}^T,$$

it follows that the matrix  $D_y(C^T C) - (C^T C) D_y$  has rank 2, hence the matrix  $C^T C$  has a displacement structure. This result leads to a fast algorithm for computing the  $QR$  factorization of a real Cauchy-like matrix by using normal equations. In fact a  $QR$  factorization of  $C$  is obtained from the Cholesky factorization of  $C^T C = R^T R$  by setting  $Q = CR^{-1}$ . By virtue of the displacement structure of  $C^T C$ , the factors  $Q$  and  $R$  can be computed in a fast way by using only  $O(n^2)$  arithmetic operations by means of the technique devised in [17].

However, it is well known that the use of normal equations yields serious numerical drawbacks since the accuracy of the computed matrix  $R$  depends on the square of the condition number of  $C$ , which can be very large [7]. Instead, a fast method for the  $QR$  factorization of Cauchy matrices was devised in [5] which does not require

to form the matrix  $C^T C$ . More specifically, the techniques devised in the above mentioned paper apply to classical Cauchy matrices defined by  $v_i = w_i = 1$  that satisfy an additional condition concerning the nonsingularity of a certain related matrix  $H$ .

Many problems on structured matrices can be solved via transformations to other classes where a problem under study already has a solution. This approach has been developed e.g., in [8, 14], where transformations of the form  $A \mapsto X_1 A X_2$  are exploited in order to convert matrices whose structure is related to one displacement equation (in particular, Toeplitz or Toeplitz-plus-Hankel matrices) into other displacement structured classes (in particular, generalized Cauchy matrices). Indeed, suppose that the matrix  $A$  is such that  $U A - A V$  has rank one, and that  $U$ ,  $V$  admit diagonalizations  $U = X_1 D_x X_1^{-1}$  and  $V = X_2 D_y X_2^{-1}$ , where the diagonal entries of  $D_x$  and  $D_y$  are all real and distinct. Then the matrix  $C = X_1^{-1} A X_2$  is a Cauchy-like matrix as in (1.1), since  $D_x C - C D_y$  has rank one. In the present context, this transformation approach would lead to the following result: If  $X_1$  is orthogonal and  $X_2$  is upper triangular (e.g., the identity), then any algorithm to compute a factorization  $C = QR$  would lead to a  $QR$  factorization of  $A$ , via the equation  $A = (X_1 Q)(R X_2^{-1})$ ; and conversely, any algorithm to compute a factorization  $A = QR$  leads to the orthogonal factorization  $C = (X_1^{-1} Q)(R X_2)$ . On the other hand, the computation of  $QR$  factorizations of structured (in particular, Toeplitz-like) matrices has already dealt with in [16] (see also Chapter 1 in [18]) as an application of the generalized Schur algorithm: The  $QR$  factorization of a matrix having a Toeplitz-like displacement structure can be computed from the  $LU$  factorization of a properly defined block matrix. By the way, the resulting algorithm are unstable and memory expensive.

In the present paper we generalize the approach followed in [5] along two different lines. Firstly, we extend these results to Cauchy-like matrices, which are better suited to capture the structure of the associated approximation problems. Secondly, we remove any additional assumption on the matrix  $C$  and this achievement is important for numerical purposes since it prevents the occurrence of near degenerate situations. The derivation of our fast method basically follows from the displacement equation (1.4), with no recourse to the transformation approach. In particular, we find that the matrices  $Q$  and  $R$  can be determined column by column and row by row, respectively, by means of suitable generalized recurrence relations at the overall cost of  $O(n^2)$  arithmetic operations. The resulting computational schemes are quite similar to Lanczos's variants for the  $QR$  factorization of Vandermonde and Vandermonde-like matrices [22]; therefore, as we have already done for their specializations of [5], they are named Lanczos-type algorithms for the  $QR$  factorization of Cauchy-like matrices.

The paper is organized in the following way. In Section 2 we outline a relationship between Cauchy-like matrices and diagonal-plus-semiseparable matrices. On the basis of this link, we derive suitable recurrence relations among the columns of the matrices  $Q$  and  $R^T$  arising in the factorization  $C = QR$ . This provides the bases of our algorithm which is presented and analyzed in Section 3. Incidentally, as a by-product of our analysis, we find a class of Cauchy-like matrices for which the polar decomposition can explicitly be computed. We remark that the only other nontrivial class of matrices which is known to admit an explicitly computable polar

decomposition is that of the Frobenius (companion) matrices, see [3]. Conclusions and possible developments are finally drawn in Section 4.

## 2. A link between Cauchy-like and semiseparable matrices

Under the assumptions stated at the beginning of this paper, it can easily be shown that the matrix  $C$  in (1.1) is invertible. Indeed, let  $D_v = \text{Diag}[v_1, \dots, v_n]$  and  $D_w = \text{Diag}[w_1, \dots, w_n]$ . Then  $C = D_w K D_v$ , where  $K \equiv (1/(x_i - y_j))$  is a classical *Cauchy matrix*. Hence  $\det(C) = \det(D_w) \det(K) \det(D_v) \neq 0$ , because of the well known nonsingularity of  $K$ , see e.g., [2, 9].

The starting point of our investigation is the following theorem, which generalizes to Cauchy-like matrices a result valid for Cauchy matrices [5, Thm. 1]:

**THEOREM 2.1.** *Let  $C = QR$ , where  $Q$  is orthogonal and  $R$  is upper triangular, be a QR factorization of the Cauchy-like matrix  $C$  defined by (1.1). Then, there exist two suitable  $n$ -vectors  $\mathbf{a} = [a_1, \dots, a_n]^T$  and  $\mathbf{b} = [b_1, \dots, b_n]^T$  such that*

$$(2.1) \quad Q^T D_x Q = D_y + \text{Diag}[a_1 b_1, \dots, a_n b_n] + \text{Triu}(\mathbf{a}\mathbf{b}^T) + \text{Tril}(\mathbf{b}\mathbf{a}^T).$$

Here  $\text{Triu}(A)$  denotes the strictly upper-triangular portion of the matrix  $A$ , where all entries on and below the main diagonal are set to zero and, similarly,  $\text{Tril}(A)$  denotes the strictly lower triangular part of  $A$ .

**PROOF.** Replacing  $C = QR$  into the displacement equation (1.4) yields

$$Q^T D_x Q = R D_y R^{-1} + Q^T \mathbf{w} \mathbf{v}^T R^{-1}.$$

The invertibility of  $R$  follows from that of  $C$ . Since  $Q^T D_x Q$  is a symmetric matrix we obtain

$$\begin{aligned} [\text{Tril}(Q^T \mathbf{w} \mathbf{v}^T R^{-1})]^T &= [\text{Tril}(Q^T D_x Q)]^T \\ &= \text{Triu}(Q^T D_x Q) \\ &= \text{Triu}(Q^T \mathbf{w} \mathbf{v}^T R^{-1}) + \text{Triu}(R D_y R^{-1}). \end{aligned}$$

Hence, from  $R D_y R^{-1} = D_y + \text{Triu}(R D_y R^{-1})$ , we have

$$Q^T D_x Q = D_y + Q^T \mathbf{w} \mathbf{v}^T R^{-1} + [\text{Tril}(Q^T \mathbf{w} \mathbf{v}^T R^{-1})]^T - \text{Triu}(Q^T \mathbf{w} \mathbf{v}^T R^{-1}).$$

and the thesis follows by setting  $\mathbf{a}^T = \mathbf{v}^T R^{-1}$  and  $\mathbf{b} = Q^T \mathbf{w}$ .  $\square$

The matrix  $H = \text{Diag}[a_1 b_1, \dots, a_n b_n] + \text{Triu}(\mathbf{a}\mathbf{b}^T) + \text{Tril}(\mathbf{b}\mathbf{a}^T)$ , which appears in the right hand side of (2.1),

$$(2.2) \quad H = \begin{bmatrix} a_1 b_1 & a_1 b_2 & \cdots & a_1 b_n \\ a_1 b_2 & a_2 b_2 & \cdots & a_2 b_n \\ \vdots & \vdots & \ddots & \vdots \\ a_1 b_n & a_2 b_n & \cdots & a_n b_n \end{bmatrix},$$

belongs to the class of symmetric *semiseparable matrices* [4, 19, 20, 23]. The next result provides a property of  $H$  which restricts the class of semiseparable matrices admissible in the present context.

**THEOREM 2.2.** *All columns of the matrix  $H = Q^T D_x Q - D_y$  are nonzero vectors. In particular,  $a_1 \neq 0$  and  $b_n \neq 0$ .*

PROOF. If  $H\mathbf{e}_j = 0$  for some index  $j$  then, from (2.1) it follows  $Q^T D_x Q \mathbf{e}_j = y_j \mathbf{e}_j$ . Hence,  $y_j$  would be an eigenvalue of  $Q^T D_x Q$ , which is not possible, since the eigenvalues of  $Q^T D_x Q$  are  $x_1, \dots, x_n$  and  $y_j \neq x_i$ .  $\square$

In the next theorem we show a generalization of classical results on the structure of the inverse of semiseparable matrices [19, 23]. This result provides the bases on which our method for the  $QR$  factorization of Cauchy-like matrices relies.

THEOREM 2.3. *Let  $H$  be given as in (2.2) and set*

$$(2.3) \quad \alpha_i = a_{i+1} b_i - a_i b_{i+1}, \quad 1 \leq i \leq n-1.$$

*Then,  $\det H = a_1 b_n \alpha_1 \alpha_2 \cdots \alpha_{n-1}$ . Let  $k$  be the number of nonzero  $\alpha_i$ , and let  $1 \leq m_i \leq n-1$  be their indices arranged in increasing order:*

$$m_0 = 0 < m_1 < \dots < m_k < n = m_{k+1}, \quad \alpha_{m_i} \neq 0.$$

*Analogously, let  $0 < n_1 < \dots < n_{n-k-1} < n$  be the indices of the possibly vanishing  $\alpha_i$ . Set  $l_i = m_i - m_{i-1}$ , for  $1 \leq i \leq k+1$ . Finally, denote by  $Z = (z_{i,j})$  the  $n \times n$  matrix such that  $z_{n_i, n_{i+1}} = 1$  for  $1 \leq i \leq n-k-1$  and  $z_{i,j} = 0$  otherwise. Then, the matrix  $H + Z$  is nonsingular and its inverse is a block tridiagonal matrix,*

$$(2.4) \quad (H + Z)^{-1} = T = \begin{bmatrix} T_{1,1} & T_{1,2} & & O \\ T_{2,1} & T_{2,2} & \ddots & \\ \ddots & \ddots & \ddots & T_{k,k+1} \\ O & & T_{k+1,k} & T_{k+1,k+1} \end{bmatrix},$$

*whose nonzero blocks have the following inner structure:*

$$T_{i,i} = \begin{bmatrix} \mu_1^{(i)} & \dots & \dots & \mu_{l_i-1}^{(i)} & \beta_i \\ 1 & 0 & \dots & 0 & \nu_1^{(i)} \\ 0 & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots \\ 0 & \dots & 0 & 1 & \nu_{l_i-1}^{(i)} \end{bmatrix} \in \mathbb{R}^{l_i \times l_i},$$

$$T_{i+1,i} = \begin{bmatrix} & \gamma_i \\ O & \end{bmatrix} \in \mathbb{R}^{l_{i+1} \times l_i}, \quad T_{i,i+1} = \begin{bmatrix} & \gamma_i \\ O & \end{bmatrix} \in \mathbb{R}^{l_i \times l_{i+1}},$$

*and  $\beta_i$ ,  $\gamma_i$ ,  $\mu_j^{(i)}$  and  $\nu_j^{(i)}$  stand for generic nonzero entries. In particular, if  $k = n-1$ ,  $H$  is nonsingular and*

$$H^{-1} = \begin{bmatrix} \beta_1 & \gamma_1 & & O \\ \gamma_1 & \beta_2 & \ddots & \\ \ddots & \ddots & \ddots & \gamma_{n-1} \\ O & \gamma_{n-1} & \beta_n & \end{bmatrix}.$$

A proof of the preceding theorem can be found in [6], except for the fact that, under the hypothesis of symmetry of  $H$ , the nonzero entries in the submatrices  $T_{i+1,i}$  and  $T_{i,i+1}$  are equal. This fact can be shown by means of the Cramer formula

for the entries of a matrix inverse. As an illustration, if  $n = 5$  and only  $\alpha_3 = 0$ , then

$$Z = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \quad (H + Z)^{-1} = \begin{pmatrix} \beta_1 & \gamma_1 & & & \\ \gamma_1 & \beta_2 & & & \\ & \gamma_2 & \mu_1^{(3)} & \beta_3 & \gamma_3 \\ & 1 & \nu_1^{(3)} & & \\ & & \gamma_3 & & \beta_4 \end{pmatrix}.$$

### 3. Derivation of the algorithm

Here and in what follows, we consider a generic Cauchy-like matrix  $C$  as in (1.1), let  $C = QR$  be its  $QR$  factorization and  $H = Q^T D_x Q - D_y$  be the semiseparable matrix (2.2). In this section, we derive our algorithm for computing  $Q$  and  $R$ . Extending a terminology introduced in [5], we call *regular* a Cauchy-like matrix  $C$  whose associated matrix  $H$  is nonsingular.

In the above mentioned paper we dealt with the  $QR$  factorization of Cauchy matrices, i.e., all the parameters  $v_i$  and  $w_i$  in (1.1) are equal to one, under the auxiliary assumption that  $C$  is regular. However, the occurrence of singularity in  $H$  was already pointed out in the above mentioned paper, with simple  $2 \times 2$  cases. Here, we will derive our algorithm for the  $QR$  factorization of the Cauchy-like matrix  $C$  of (1.1) under the sole assumptions stated at the beginning of this paper. For the sake of simplicity, the factorization scheme is firstly described for input regular Cauchy-like matrices and, then, it is extended to the general case.

In view of Theorems 2.2 and 2.3, a necessary and sufficient condition for  $C$  to be regular is  $\alpha_i \neq 0$ , for  $1 \leq i \leq n-1$ . The next result shows that this condition can be suitably reformulated in terms of the entries of the upper triangular factor  $R$ . Indeed, the occurrence of zeros in the sequence  $\alpha_1, \dots, \alpha_{n-1}$  implies the appearance of a particular pattern of zeros in the matrix  $R$  like the one figuratively shown in the next picture:

$$(3.1) \quad R = \begin{pmatrix} \ddots & * & * & \cdots & * & * & \cdots \\ * & 0 & \cdots & 0 & * & \cdots \\ & * & \ddots & \vdots & \vdots \\ & & \ddots & 0 & \vdots \\ & & & * & * & \cdots \\ & & & & * & \\ & & & & & \ddots \end{pmatrix}.$$

This fact is established by the subsequent theorem, whose proof can be found in [5]:

**THEOREM 3.1.** *Let  $\alpha_i$  be given as in (2.3). Then,  $\alpha_i = \dots = \alpha_{i+l} = 0$  if and only if  $r_{i,i+1} = \dots = r_{i,i+l+1} = 0$  where  $R \equiv (r_{i,j})$  denotes the triangular factor of the  $QR$  factorization of  $C$ . Moreover, in this case,  $r_{i+j,i+k+1} = 0$  for all indices  $j, k$  such that  $0 \leq j \leq k \leq l$ .*

The (block) tridiagonal structure of  $T = (H + Z)^{-1}$  shown in Theorem 2.3 gives the basis for the efficient recursive computation of the columns of  $Q$  and  $R^T$ .

Indeed, rewriting (2.1) as  $D_x Q - Q D_y = Q(H + Z - Z)$  and multiplying from the right by  $T$ , we obtain immediately

$$(3.2) \quad (D_x Q - Q D_y)T = Q(I - ZT).$$

Consider the integers  $k$ ,  $0 = m_0 < m_1 < \dots < m_k < m_{k+1} = n$ , and  $l_i = m_i - m_{i-1}$ , defined in Theorem 2.3. Observe that the matrix  $I - ZT$  has a block diagonal structure,

$$I - ZT = \begin{pmatrix} \Delta_1 & & O \\ & \ddots & \\ O & & \Delta_{k+1} \end{pmatrix},$$

where  $\Delta_i \in \mathbb{R}^{l_i \times l_i}$ ,  $\Delta_i = 1$  if  $l_i = 1$  and

$$\Delta_i = \begin{pmatrix} 0 & \dots & 0 & -\nu_1^{(i)} \\ \vdots & & \vdots & \vdots \\ 0 & \dots & 0 & -\nu_{l_i-1}^{(i)} \\ 0 & \dots & 0 & 1 \end{pmatrix}$$

if  $l_i > 1$ . Consider the  $m_i$ -th column of equation (3.2), for  $1 \leq i \leq k$ . Since the  $m_i$ -th column of  $T$  is

$$[0, \dots, 0, \gamma_{i-1}, \overbrace{0, \dots, 0}^{l_{i-1}-1}, \beta_i, \nu_1^{(i)}, \dots, \nu_{l_i-1}^{(i)}, \gamma_i, 0, \dots, 0]^T,$$

if we let  $\mathbf{q}_i = Q\mathbf{e}_i$ ,  $\gamma_0 = 0$  and  $\gamma_{k+1} = 0$ , using Theorem 2.3 we have

$$\begin{aligned} (D_x Q - Q D_y) \left( \gamma_{i-1} \mathbf{e}_{m_{i-2}+1} + \beta_i \mathbf{e}_{m_{i-1}+1} + \gamma_i \mathbf{e}_{m_i+1} + \sum_{j=1}^{l_i-1} \nu_j^{(i)} \mathbf{e}_{m_{i-1}+j+1} \right) &= \\ &= \mathbf{q}_{m_i} - \sum_{j=1}^{l_i-1} \nu_j^{(i)} \mathbf{q}_{m_{i-1}+j}. \end{aligned}$$

All terms containing  $\nu_j^{(i)}$  in the preceding equations vanish when  $l_i = 1$ . For simplicity of notation, introduce the vectors

$$(3.3) \quad \begin{aligned} \mathbf{s}_i &= \mathbf{q}_{m_i} - \gamma_{i-1} (D_x - y_{m_{i-2}+1} I) \mathbf{q}_{m_{i-2}+1} - \\ &\quad - \sum_{j=1}^{l_i-1} \nu_j^{(i)} [\mathbf{q}_{m_{i-1}+j} + (D_x - y_{m_{i-1}+j+1} I) \mathbf{q}_{m_{i-1}+j+1}]. \end{aligned}$$

After simple passages, we obtain the following recurrence among the columns of  $Q$ :

$$(3.4) \quad \mathbf{q}_{m_i+1} = \frac{1}{\gamma_i} (D_x - y_{m_i+1} I)^{-1} \{ \mathbf{s}_i - \beta_i (D_x - y_{m_{i-1}+1} I) \mathbf{q}_{m_{i-1}+1} \}.$$

From the orthogonality relation  $\mathbf{q}_{m_i}^T \mathbf{q}_{m_i+1} = 0$ , we obtain a formula for  $\beta_i$ :

$$(3.5) \quad \beta_i = \frac{\mathbf{q}_{m_i}^T (D_x - y_{m_i+1} I)^{-1} \mathbf{s}_i}{\mathbf{q}_{m_i}^T (D_x - y_{m_i+1} I)^{-1} (D_x - y_{m_{i-1}+1} I) \mathbf{q}_{m_{i-1}+1}}.$$

Remark that  $\mathbf{q}_1$  can be obtained by normalizing the first column of  $C$  in the 2-norm, and the coefficient  $\gamma_i$  can be obtained by normalizing the 2-norm of the right hand side of (3.4).

In what follows, we derive analogous recurrences for the columns of  $R^T$ . Premultiplying (3.2) by  $C^T$  we have

$$C^T(D_x Q - Q D_y)T = C^T Q(I - ZT) = R^T(I - ZT).$$

From (1.5) we have  $C^T D_x = \mathbf{v} \mathbf{w}^T + D_y C^T$ , and recalling that  $\mathbf{w}^T Q = \mathbf{b}^T$  in view of Theorem 2.1, we obtain

$$\begin{aligned} C^T(D_x Q - Q D_y)T &= (D_y R^T + \mathbf{v} \mathbf{w}^T Q - R^T D_y)T \\ &= (D_y R^T - R^T D_y)T + \mathbf{v} \mathbf{b}^T T \\ &= (D_y R^T - R^T D_y)T + \frac{1}{a_1} \mathbf{v} \mathbf{e}_1^T (H + Z - Z)T \\ &= (D_y R^T - R^T D_y)T + \frac{1}{a_1} \mathbf{v} \mathbf{e}_1^T (I - ZT). \end{aligned}$$

In the preceding passages, we used the equation  $\mathbf{e}_1^T H = a_1 \mathbf{b}^T$ , which is apparent from (2.2). Again from Theorem 2.1, we have  $\mathbf{a}^T = \mathbf{v}^T R^{-1}$ , hence  $a_1 = v_1/r_{1,1}$ . Recall that  $a_1 \neq 0$  because of Theorem 2.2. Finally we have

$$(3.6) \quad (D_y R^T - R^T D_y)T = (R^T - \frac{r_{1,1}}{v_1} \mathbf{v} \mathbf{e}_1^T)(I - ZT).$$

Consider the  $m_i$ -th column of the above equation. Denoting  $\mathbf{r}_i^T = \mathbf{e}_i^T R$  the  $i$ -th row of the factor  $R$ , introduce the vectors

$$\begin{aligned} (3.7) \quad \mathbf{t}_i &= \mathbf{r}_{m_i} - \gamma_{i-1}(D_y - y_{m_{i-2}+1} I)\mathbf{r}_{m_{i-2}+1} - \\ &\quad - \sum_{j=1}^{l_i-1} \nu_j^{(i)} [\mathbf{r}_{m_{i-1}+j} + (D_y - y_{m_{i-1}+j+1} I)\mathbf{r}_{m_{i-1}+j+1}]. \end{aligned}$$

For  $2 \leq i \leq k$  we have the formula

$$(3.8) \quad (D_y - y_{m_i+1} I)\mathbf{r}_{m_i+1} = \frac{1}{\gamma_i} \{\mathbf{t}_i - \beta_i(D_y - y_{m_{i-1}+1} I)\mathbf{r}_{m_{i-1}+1}\}.$$

Furthermore, if  $m_1 = 1$  we have the equation

$$(3.9) \quad (D_y - y_2 I)\mathbf{r}_2 = \frac{1}{\gamma_1} \{\mathbf{r}_1 - \beta_1(D_y - y_1 I)\mathbf{r}_1\} - \frac{r_{1,1}}{\gamma_1 v_1} \mathbf{v},$$

while for  $m_1 > 1$  we have

$$(3.10) \quad (D_y - y_{m_1+1} I)\mathbf{r}_{m_1+1} = \frac{1}{\gamma_1} \{\mathbf{t}_1 - \beta_1(D_y - y_1 I)\mathbf{r}_1\} - \frac{\beta_1 r_{1,1}}{\gamma_1 v_1} \mathbf{v}.$$

Finally, we have  $\mathbf{r}_1 = C^T \mathbf{q}_1$ .

Since the matrix  $(D_y - y_{m_i+1} I)$  is singular, equation (3.8) does not allow to compute  $r_{m_i+1, m_i+1}$ ; indeed, the first  $m_i + 1$  entries of the vector  $(D_y - y_{m_i+1} I)\mathbf{r}_{m_i+1}$  are zero. On the other hand, the first nonzero entry of  $(D_y - y_{m_{i-1}+1} I)\mathbf{r}_{m_{i-1}+1}$  is in position  $m_i + 1$ ; hence, the first  $m_i$  entries of  $\mathbf{t}_i$  must be zero. Let  $1 \leq p \leq l_i$ . From  $\mathbf{e}_{m_{i-1}+p}^T \mathbf{t}_i = 0$  and (3.7) we have

$$\begin{aligned} &r_{m_i, m_{i-1}+p} - \gamma_{i-1}(y_{m_{i-1}+p} - y_{m_{i-2}+1})r_{m_{i-2}+1, m_{i-1}+p} = \\ &= \sum_{j=1}^p \nu_j^{(i)} [r_{m_{i-1}+j, m_{i-1}+p} + (y_{m_{i-1}+p} - y_{m_{i-1}+j+1})r_{m_{i-1}+j+1, m_{i-1}+p}]. \end{aligned}$$

Recall that the coefficients  $\nu_j^{(i)}$  disappear when  $l_i = 1$ . Taking into account the structure of  $R$  shown in (3.1), the right hand side of the above formula simplifies

to  $\nu_p^{(i)} r_{m_{i-1}+p, m_{i-1}+p}$ , when  $1 \leq p < l_i$ , and vanishes, when  $p = l_i$ . In the first case we obtain

$$(3.11) \quad r_{m_{i-1}+p, m_{i-1}+p} = -\frac{\gamma_{i-1}}{\nu_p^{(i)}} (y_{m_{i-1}+p} - y_{m_{i-2}+1}) r_{m_{i-2}+1, m_{i-1}+p},$$

while for  $p = l_i$  we have

$$(3.12) \quad r_{m_i, m_i} = \gamma_{i-1} (y_{m_i} - y_{m_{i-2}+1}) r_{m_{i-2}+1, m_i}.$$

Summing up, we arrive at the the following algorithm to compute the QR factorization of the matrix  $C$ , in the case that  $H$  is nonsingular. In fact, in this case we have  $k = n - 1$ ,  $m_i = i$ , and all terms involving  $\nu_j^{(i)}$  disappear.

**Algorithm:** QR factorization of a regular Cauchy-like matrix  $C = [\mathbf{c}_1 | \dots | \mathbf{c}_n]$ .

**Input:** The parameters  $x, y, v$  and  $w$  defining the matrix  $C$  in (1.1).

**Output:** An orthogonal matrix  $Q = [\mathbf{q}_1 | \dots | \mathbf{q}_n]$  and an upper triangular matrix  $R \equiv (r_{i,j})$ ,  $R^T = [\mathbf{r}_1 | \dots | \mathbf{r}_n]$ , such that  $C = QR$ .

**Computation:**

Set  $\mathbf{q}_0 = 0$  and  $\gamma_0 = 0$ .

Compute  $r_{1,1} = \|\mathbf{c}_1\|_2$ ,  $\mathbf{q}_1 = \mathbf{c}_1/r_{1,1}$  and  $\mathbf{r}_1 = C^T \mathbf{q}_1$ .

**For**  $i = 1, \dots, n - 1$  **do**

    Compute  $\mathbf{s}_i$  from equation (3.3).

    Compute  $\beta_i$  from (3.5).

    Compute  $\mathbf{z} = (D_x - y_{i+1})^{-1} \{ \mathbf{s}_i - \beta_i (D_x - y_i I) \mathbf{q}_i \}$ .

    Compute  $\gamma_i = \|\mathbf{z}\|_2$ .

    Compute  $\mathbf{q}_{i+1} = \mathbf{z}/\gamma_i$ .

    Compute  $r_{i+1, i+1}$  from (3.12) and  $r_{i+1, i+2}, \dots, r_{i+1, n}$  via equation (3.8) or (3.9).

**End.**

The preceding algorithm does not work properly when  $H$  is singular, hence when some of the  $\alpha_i$  in (2.3) vanish. We can detect this fact by simply looking at the strictly upper part of  $R$ : Suppose that, for some integer  $1 \leq i \leq k + 1$ , we have  $l_i \geq 2$ . For simplicity of notations, in what follows let  $p = m_{i-1} + 1$  and  $q = m_i - 1$ . Then, for  $p \leq j \leq q$  we have  $\alpha_j = 0$  while  $\alpha_{p-1} \neq 0$  and  $\alpha_{q+1} \neq 0$  (we can set  $\alpha_0$  and  $\alpha_n$  to any nonzero number). Moreover, from Theorem 3.1,  $r_{p,j+1} = 0$ . Redefine  $\mu_1^{(i)}, \dots, \mu_{l_i-1}^{(i)}$  and  $\nu_1^{(i)}, \dots, \nu_{l_i-1}^{(i)}$  as  $\mu_p, \dots, \mu_q$  and  $\nu_p, \dots, \nu_q$ , respectively. Then, from (3.2) and the structure of  $T$  described in Theorem 2.3 we obtain

$$(D_x Q - Q D_y)(\mathbf{e}_{j+1} + \mu_j \mathbf{e}_p) = 0, \quad p \leq j \leq q.$$

We obtain

$$(D_x - y_{j+1} I) \mathbf{q}_{j+1} = -\mu_j (D_x - y_p I) \mathbf{q}_p.$$

Then, for  $j = p, \dots, q$  we replace the relation (3.4) with the following:

$$(3.13) \quad \mathbf{q}_{j+1} = -\mu_j (D_x - y_{j+1} I)^{-1} (D_x - y_p I) \mathbf{q}_p.$$

Proceeding analogously for the columns of  $R^T$ , we find from (3.6)

$$(D_y R^T - R^T D_y)(\mathbf{e}_{j+1} + \mu_j \mathbf{e}_p) = 0, \quad p \leq j \leq q.$$

We obtain

$$(3.14) \quad (D_y - y_{j+1} I) \mathbf{r}_{j+1} = -\mu_j (D_y - y_p I) \mathbf{r}_p.$$

As for equation (3.8), the preceding equation can only be used to compute the off-diagonal entries of  $\mathbf{r}_{j+1}$ . The diagonal entries of  $R$  can be computed from (3.12) and (3.11), which require the knowledge of the coefficients  $\nu_j$ . These numbers can be computed by means of the following argument: From (2.1) we have  $Q^T D_x - D_y Q^T = (H + Z - Z)Q^T$ . Premultiplying by  $T$  and trasposing we obtain, in parallel with (3.2),

$$(D_x Q - Q D_y) T^T = Q(I - Z^T T^T).$$

The matrix  $I - Z^T T^T$  is block diagonal,

$$I - Z^T T^T = \begin{pmatrix} \hat{\Delta}_1 & & O \\ & \ddots & \\ O & & \hat{\Delta}_{k+1} \end{pmatrix},$$

where  $\hat{\Delta}_i \in \mathbb{R}^{l_i \times l_i}$ ,  $\hat{\Delta}_i = 1$  if  $l_i = 1$  and

$$\hat{\Delta}_i = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ -\mu_p & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ -\mu_q & 0 & \cdots & 0 \end{pmatrix}$$

if  $l_i > 1$ . For  $p \leq j \leq q$ , the  $(j+1)$ -th column of the preceding equation gives

$$(D_x Q - Q D_y)(\mathbf{e}_j + \nu_j \mathbf{e}_{q+1}) = 0.$$

whence we have

$$(D_x - y_j I)\mathbf{q}_j = -\nu_j(D_x - y_{q+1} I)\mathbf{q}_{q+1}, \quad p \leq j \leq q.$$

Then, if  $1 \leq h \leq n$  is any index such that  $\mathbf{e}_h^T \mathbf{q}_{q+1} \neq 0$ , we have

$$(3.15) \quad \nu_j = -\frac{(x_h - y_j)\mathbf{e}_h^T \mathbf{q}_j}{(x_h - y_{q+1})\mathbf{e}_h^T \mathbf{q}_{q+1}}, \quad p \leq j \leq q.$$

Thus we can recover the coefficients  $\nu_j$  by means of the latter equation from the vectors  $\mathbf{q}_p, \dots, \mathbf{q}_{q+1}$  previously computed via (3.13). We arrive at the complete algorithm:

**Algorithm:** QR factorization of a generic Cauchy-like matrix  $C = [\mathbf{c}_1 | \dots | \mathbf{c}_n]$ .

**Input:** The parameters  $x, y, v$  and  $w$  defining the matrix  $C$  in (1.1).

**Output:** An orthogonal matrix  $Q = [\mathbf{q}_1 | \dots | \mathbf{q}_n]$  and an upper triangular matrix  $R \equiv (r_{i,j})$ ,  $R^T = [\mathbf{r}_1 | \dots | \mathbf{r}_n]$ , such that  $C = QR$ .

**Computation:**

Set  $i = 1$ ,  $\mathbf{q}_0 = 0$  and  $\gamma_0 = 0$ .

Compute  $r_{1,1} = \|\mathbf{c}_1\|_2$ ,  $\mathbf{q}_1 = \mathbf{c}_1/r_{1,1}$  and  $\mathbf{r}_1 = C^T \mathbf{q}_1$ .

**While**  $i < n$  **do**

**If**  $r_{i,i+1} = 0$  **then**

Let  $k$  be the smallest integer greater than  $i$  such that  $r_{i,k} = 0$  and  $r_{i,k+1} \neq 0$  (if  $r_{i,i+1} = \dots = r_{i,n} = 0$  set  $k = n$ ).

**For**  $j = i+1, \dots, k$  **do**

Let  $\mathbf{z} = (D_x - y_j)^{-1}(D_x - y_i)\mathbf{q}_i$ .

Compute  $\mu_j = 1/\|\mathbf{z}\|_2$ .

Compute  $\mathbf{q}_j = -\mu_j \mathbf{z}$ .

```

End
For  $j = i + 1, \dots, k$  do
    Compute  $\nu_j$  from (3.15).
    Compute  $r_{j,j}$  from equations (3.12) or (3.11), set  $r_{j,j+1} = \dots = r_{j,k} = 0$  and compute  $r_{j,k+1}, \dots, r_{j,n}$  from (3.14).
End
Else set  $k = i$ .
End
Set  $l = i$  and  $i = k + 1$ .
If  $i = n$  then Stop.
Compute  $s_i$  from equation (3.3).
Compute  $\beta_i$  from (3.5).
Compute  $\mathbf{z} = (D_x - y_{i+1})^{-1}\{\mathbf{s}_i - \beta_i(D_x - y_l I)\mathbf{q}_l\}$ .
Compute  $\gamma_i = \|\mathbf{z}\|_2$ .
Compute  $\mathbf{q}_{i+1} = \mathbf{z}/\gamma_i$ .
Compute  $r_{i+1,i+1}$  from (3.12) and  $r_{i+1,i+2}, \dots, r_{i+1,n}$  from (3.8–3.10).
End.

```

This algorithm can be implemented in  $O(n^2)$  multiplications and additions. Moreover, besides to the matrices  $Q$  and  $R$ , only  $O(n)$  memory space is needed to store auxiliary variables.

Finally, from Theorem 3.1, we observe that, when  $\alpha_1 = \dots = \alpha_{n-1} = 0$  the matrix  $R$  becomes diagonal. In this case, if we construct the factors  $Q$  and  $R$  so that  $r_{i,i} > 0$ , the factorization  $C = QR$  becomes the polar decomposition. We remark that the only other nontrivial class of matrices known to admit an explicit polar decomposition is that of Frobenius (companion) matrices, see [3].

The following result allows one to check if the polar decomposition of the matrix (1.1) can be computed explicitly:

**COROLLARY 3.2.** *If for  $j = 2, \dots, n$  one has*

$$\sum_{i=1}^n \frac{w_i^2}{(x_i - y_1)(x_i - y_j)} = 0,$$

*then the polar factorization of  $C$  can be computed explicitly.*

**PROOF.** If we denote by  $\mathbf{c}_1$  the first column of  $C$ , the stated hypothesis means that  $\mathbf{c}_1^T C$  is a multiple of  $\mathbf{e}_1^T$ . If this is the case, since  $\mathbf{c}_1^T C = \mathbf{e}_1^T C^T C = \mathbf{e}_1^T R^T R = r_{1,1} \mathbf{e}_1^T R$ , then the first row of the triangular factor  $R$  has the form  $[r_{1,1}, 0, \dots, 0]$ . In the light of Theorem 3.1 we conclude that all the parameters  $\alpha_i$  vanish and the matrix  $R$  is diagonal. Hence, the orthogonal matrix  $Q$  can be obtained by simply normalizing the 2-norm of the columns of  $C$ .  $\square$

According to the notations used in the Introduction, the hypothesis of the preceding corollary can be restated as  $\langle \phi_1, \phi_j \rangle = 0$ , for  $2 \leq j \leq n$ . If the latter is true, the factors  $Q = [\mathbf{q}_1 | \dots | \mathbf{q}_n]$  and  $D = \text{Diag}[d_1, \dots, d_n]$  of the polar decomposition  $C = QD$  can be computed by means of the following steps:

```

For  $i = 1, \dots, n$  do
    Compute  $d_i = \|\mathbf{c}_i\|_2$ .
    Set  $\mathbf{q}_i = \mathbf{c}_i/d_i$ .
End.

```

Altough the existence of Cauchy-like matrices fulfilling the hypothesis of Corollary 3.2 is suggested by numerical evidence, we are not able to explicitly construct them. However, we note that a necessary condition for the validity of the hypothesis of Corollary 3.2 is that the nodes  $x_i$  and  $y_i$  are interlaced, for example,  $x_1 < y_1 < x_2 < \dots < y_n$ , or  $y_1 < x_1 < y_2 < \dots < x_n$ . Indeed, if this is not the case, two columns of  $C$  have equal (or complementary) sign patterns, hence they cannot be orthogonal.

#### 4. Conclusions and further developments

In this paper we have developed an algorithm for computing a  $QR$  factorization of Cauchy-like matrices  $C$  with real nodes. The proposed scheme is based on the existence of certain generalized recurrences among the columns of  $Q$  and  $R^T$ . These recurrences allow a recursive construction of these matrices using  $O(n^2)$  arithmetic operations without never forming the matrix  $C^TC$ . Furthermore, we have shown that the possible singularity of  $H = Q^TD_xQ - D_y$  induces a suitable block structure of the triangular factor  $R$ . By exploiting this structure, we are also able to identify a special class of Cauchy-like matrices admitting an explicitly computable polar decomposition.

Our approach proceeds quite closely with the theory on Lanczos-type algorithms for the fast solution of Vandermonde-like linear systems. In view of this similarity, we believe that many results and constructions concerning the theory of classical orthogonal polynomials which are derived using matrix arguments involving Vandermonde-like matrices could be generalized to families of orthogonal rational functions by replacing Vandermonde-like matrices with Cauchy-like ones.

Our numerical experience with the algorithms developed here, indicates a poor behavior of our method when compared with the performance of classical  $O(n^3)$  algorithms [10], where the orthogonal factor  $Q$  is formed as a product of elementary orthogonal matrices. This is indeed a quite general situation and an interesting discussion on the difficulties in devising fast and stable algorithms for the  $QR$  factorization of structured (Toeplitz-like) matrices can be found in [16] and in the book [18]. Certainly, one should explore the possibility to improve numerical stability by devising other formulas to compute the recurrence coefficients.

A quite different perspective to the computation of  $C = QR$  that can achieve better numerical properties relies upon Theorem 2.1. Indeed, that theorem exhibits the spectral decomposition of a diagonal plus a symmetric semiseparable matrix: hence the task of computing the factorization  $C = QR$  can therefore be reformulated as an inverse eigenvalue problem for diagonal plus symmetric semiseparable matrices, that is, to determine the semiseparable matrix  $H$  so that the eigenvalues of  $D_y + H$  are  $x_1, \dots, x_n$ . Following this approach, a very interesting task is that of suitably modifying the algorithm proposed by Rutishauser, Gragg and Harrod in [11] for the solution of an inverse eigenvalue problem for a tridiagonal symmetric matrix in order to cope with the structure of  $H$  devised in (2.2). Indeed, a method for the fast  $QR$  factorization of Vandermonde-like matrices based on the Rutishauser, Gragg and Harrod algorithm was found to generally yield higher accuracy than a certain Lanczos-type (Stieltjes) procedure [22].

## References

- [1] E. Hendriksen A. Bultheel, P. González-Vera and O. Njåstad. *Orthogonal rational functions*, volume 5 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, 1999.
- [2] D. Calvetti and L. Reichel. On the solution of Cauchy systems of equations. *Electronic Transactions on Numerical Analysis*, 4:125–137, 1996.
- [3] P. van den Driessche and H. K. Wimmer. Explicit polar decomposition of companion matrices. *Electron. J. Linear Algebra* 1 (1996), 64–69.
- [4] Y. Eidelman and I. Gohberg. Fast inversion algorithms for diagonal plus semiseparable matrices. *Integral Equations Operator Theory*, 27:165–183, 1997.
- [5] D. Fasino and L. Gemignani. A Lanczos-type algorithm for the QR factorization of regular Cauchy matrices. *Numer. Lin. Algebra Appl.*, to appear, 2002.
- [6] D. Fasino and L. Gemignani. Structural and computational properties of possibly singular semiseparable matrices. *Linear Algebra Appl.*, 340:183–198, 2002.
- [7] D. Fasino and V. Olshevsky. How bad are symmetric Pick matrices? *Structured Matrices in Mathematics, Computer Science, and Engineering, I*, Contemp. Math., 280:301–311, Amer. Math. Soc., Providence, RI, 2001.
- [8] I. Gohberg, T. Kailath, and V. Olshevsky. Fast Gaussian elimination with partial pivoting for matrices with displacement structure. *Mathematics of Computation*, 64:1557–1576, 1995.
- [9] I. Gohberg and I. Koltracht; On the inversion of Cauchy matrices, in: *Signal processing, scattering and operator theory, and numerical methods* (Amsterdam, 1989), 381–392, Progr. Systems Control Theory, 5, Birkhäuser Boston, Boston, MA, 1990.
- [10] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The John Hopkins University Press, Baltimore, Maryland, 1989.
- [11] W. B. Gragg and W. J. Harrod. The numerically stable reconstruction of Jacobi matrices from spectral data. *Numerische Mathematik*, 44:317–335, 1984.
- [12] M. Gu. Stable and efficient algorithms for structured systems of linear equations. *SIAM J. Matrix Anal. Appl.*, 19:279–306, 1998.
- [13] G. Heinig. Inversion of generalized Cauchy matrices and other classes of structured matrices. *Linear algebra for signal processing*, IMA Vol. Math. Appl., 69:63–81, Springer, New York, 1995.
- [14] G. Heinig and A. Bojanczyk. Transformation techniques for Toeplitz and Toeplitz-plus-Hankel matrices. II. Algorithms. *Linear Algebra Appl.*, 278:11–36, 1998.
- [15] P. Junghanns and D. Oestreich. Numerische Lösung des Staudammproblems mit Drainage. *Zeitschrift für Angewandte Mathematik und Mechanik*, 69:83–92, 1989.
- [16] T. Kailath and J. Chun. Generalized displacement structure for block-Toeplitz, Toeplitz-block, and Toeplitz-derived matrices. *SIAM J. Matrix Anal. Appl.*, 15:114–128, 1994.
- [17] T. Kailath and V. Olshevsky. Diagonal pivoting for partially reconstructible Cauchy-like matrices, with applications to Toeplitz-like linear equations and to boundary rational matrix interpolation problems. *Linear Algebra Appl.*, 254:251–302, 1997.
- [18] T. Kailath and A. H. Sayed, editors. *Fast reliable algorithms for matrices with structure*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1999.
- [19] I. Koltracht. Linear complexity algorithm for semiseparable matrices. *Integral Equations Operator Theory*, 29:313–319, 1997.
- [20] N. Mastronardi, S. Chandrasekaran, and S. van Huffel. Fast and stable reduction of diagonal plus semi-separable matrices to tridiagonal and bidiagonal form. *BIT*, 41:149–157, 2001.
- [21] V. Olshevsky and V. Y. Pan. Polynomial and rational evaluation and interpolation (with structured matrices). *Automata, languages and programming*, 585–594, Lecture Notes in Comput. Sci., 1644, Springer, Berlin, 1999.
- [22] L. Reichel. Fast QR decomposition of Vandermonde-like matrices and polynomial least squares approximation. *SIAM J. Matrix Anal. Appl.*, 12:552–564, 1991.
- [23] P. Rózsa, R. Bevilacqua, F. Romani, and P. Favati. On band matrices and their inverses. *Linear Algebra Appl.*, 150:287–295, 1991.
- [24] E. E. Tyrtyshnikov. Cauchy-Toeplitz matrices and some applications. *Linear Algebra and Its Applications*, 149:1–18, 1991.

DIPARTIMENTO DI MATEMATICA E INFORMATICA, UNIVERSITÀ DI UDINE, 33100 UDINE, ITALY  
*E-mail address:* `fasino@dimi.uniud.it`

DIPARTIMENTO DI MATEMATICA, UNIVERSITÀ DI PISA, 56127 PISA, ITALY  
*E-mail address:* `gemignan@dm.unipi.it`

# Fast and Stable Algorithms for Reducing Diagonal plus Semiseparable Matrices to Tridiagonal and Bidiagonal Form

Dario Fasino, Nicola Mastronardi, and Marc Van Barel

**ABSTRACT.** New algorithms to reduce a diagonal plus a symmetric semiseparable matrix to a symmetric tridiagonal one and an algorithm to reduce a diagonal plus an unsymmetric semiseparable matrix to a bidiagonal one are considered. The algorithms are fast and stable, requiring a computational cost of  $O(N^2)$ , where  $N$  is the order of the considered matrix.

## 1. Introduction.

Consider a  $N \times N$  matrix of the form

$$(1) \quad A = \begin{bmatrix} x_1 & u_1 v_2 & u_1 v_3 & \cdots & u_1 v_N \\ u_1 v_2 & x_2 & u_2 v_3 & \cdots & u_2 v_N \\ u_1 v_3 & u_2 v_3 & \ddots & \cdots & \cdots \\ \vdots & \vdots & \vdots & x_{N-1} & u_{N-1} v_N \\ u_1 v_N & u_2 v_N & \vdots & u_{N-1} v_N & x_N \end{bmatrix}.$$

Matrices with the above structure can be decomposed into the sum of a diagonal and a semiseparable matrix, hence they are usually called *diagonal-plus-semiseparable* (DSS) matrices [6, 12, 13]. Let  $\mathbf{u} = [u_1, \dots, u_{N-1}, u_N]^T$ ,  $\mathbf{v} = [v_1, v_2, \dots, v_N]^T$  and  $\mathbf{x} = [x_1, x_2, \dots, x_N]^T$ , where the values of  $u_N$  and  $v_1$  are irrelevant. We will use the shorthand

$$(2) \quad A = \mathcal{DSS}(\mathbf{u}, \mathbf{v}, \mathbf{x})$$

---

1991 *Mathematics Subject Classification*. Primary 15A09; Secondary 15A23, 65F05, 65L10, 65R20.

*Key words and phrases.* Diagonal plus semiseparable matrices, fast algorithms, stable algorithms.

This research was partially supported by the K.U.Leuven (Bijzonder Onderzoeksfonds), project “SLAP: Structured Linear Algebra Package,” grant #OT/00/16 and partially by the Fund for Scientific Research–Flanders (FWO–V), project “SMA: Structured Matrices and their Applications” grant #G.0078.01. The work of NM and MVB is also supported by the Belgian Programme on Interuniversity Poles of Attraction, initiated by the Belgian State, Prime Minister’s Office for Science, Technology and Culture. The scientific responsibility rests with the authors.

to denote the matrix (1). DSS matrices appear in the theory of statistics as variance–covariance matrices of ordered observations of random samples from rectangular distributions [11]. Such matrices arise also from discretizations of Green functions for two point boundary value problems, and also from the discretization of eigenvalue problems for integral operators whose kernels can be written as the sum of a semiseparable part and a degenerate one. The inverse of a symmetric tridiagonal matrix has this form too.

The standard direct procedure for computing the eigendecomposition of a symmetric  $A$  first reduces  $A$  into tridiagonal form [10, 14]. This algorithm doesn't take into account the structure of  $A$  and requires  $O(N^3)$  flops. Although Lanczos' method can take advantage of the structure of  $A$  and can be used to find the eigendecomposition of  $A$ , the convergence rate is highly problem-dependent and can be very slow in many cases [14]. The fast algorithms for tridiagonalizing diagonal plus semiseparable matrices described in [3, 4] cost  $O(N^2)$  flops. A two-way chasing version of the latter algorithms has been considered in [12], halving the computational complexity. The two-way chasing technique has been introduced in [18] and further investigated in [16] to halve the computational complexity of the algorithm proposed in [1] for reducing arrowhead matrices into tridiagonal form. Remark that also arrowhead matrices are DSS.

These matrices belong to a more general class of structured matrices as defined, e.g., in [5, 7, 8]. Besides diagonal-plus-semiseparable matrices, this class also contains the set of banded matrices and the set of unitary Hessenberg matrices. For this general class of matrices, different fast inversion and factorization algorithms were studied. Moreover, the orthogonal transformations used in Algorithm 2 coincide with those used in [8] in the special case of diagonal-plus-semiseparable matrices. However, in our paper, we use these orthogonal transformations to construct a similarity transformation for solving the eigenvalue problem while they are used in [8] to solve the corresponding system of linear equations.

In this paper we consider new algorithms to reduce symmetric diagonal plus semiseparable matrices into tridiagonal ones and diagonal plus semiseparable matrices into bidiagonal ones with  $O(N^2)$  computational complexity.

The paper is organized as follows. In §2 a new algorithm for tridiagonalizing symmetric DSS matrices is described. In §3 a two-way algorithm having half of the computational complexity is derived. In §4 a fast algorithm is presented to reduce an unsymmetric diagonal plus semiseparable matrix to a bidiagonal one. Since the latter algorithm does not accumulate the computed Givens rotations, its application is particularly recommended in the following cases:

- (1) Only the computation of the singular values is required.
- (2) One wants to solve  $\min_{\mathbf{c}} \|\mathbf{b} - B\mathbf{c}\|_2^2 + \lambda \|\mathbf{c}\|_2^2$ , for many values of  $\lambda$ , where  $B$  is a rectangular matrix made up by the first  $M \leq N$  columns of the matrix  $A$  in (1), and  $\mathbf{b}, \mathbf{c}$  are vectors of appropriate dimensions [2, 9].

Furthermore, §5 gives some numerical experiments and § 6 contains the conclusions.

## 2. Reduction of symmetric DSS matrices into tridiagonal ones

In this section, we give an algorithm to compute a tridiagonal matrix similar to the matrix  $A$  of (1). The algorithm consists in  $N - 2$  steps and works as follows: In the first step, we use some Givens rotations to reduce the original matrix  $A$  into

the following form:

$$\left[ \begin{array}{c|c} & \begin{matrix} 0 \\ \vdots \\ 0 \\ b_{N-1} \end{matrix} \\ A_1 & \\ \hline \begin{matrix} 0 & \cdots & 0 & b_{N-1} \end{matrix} & d_N \end{array} \right],$$

where  $A_1$  is DSS and has order  $N - 1$ . Then, the same procedure is applied recursively starting from  $A_1$ , until the leading block has order 2, ending up with a symmetric tridiagonal matrix,

$$(3) \quad \left[ \begin{matrix} d_1 & b_1 & & O \\ b_1 & d_2 & \ddots & \\ \ddots & \ddots & \ddots & b_{N-1} \\ O & & b_{N-1} & d_N \end{matrix} \right].$$

In what follows, we will suppose  $u_1 \neq 0$ . If this is not the case, the matrix  $A$  can be deflated or, more precisely, the problem of tridiagonalizing  $A$  can be restricted to its trailing submatrix. Let  $I_n$  denote the identity matrix of order  $n$ . Define the scalars  $f_1, \dots, f_{N-1}$  as follows:

$$(4) \quad f_1 = u_1, \quad f_j = \sqrt{\sum_{k=1}^j u_k^2}, \quad j = 2, \dots, N-1.$$

Note that  $f_i \neq 0$ , for  $1 \leq i \leq N-1$ . We describe the generic step of our algorithm as follows, using a Matlab-style notation:

### Algorithm 1

```

 $A^{(0)} = A;$ 
 $N = \max(\text{size}(A^{(0)}));$ 
for  $i = 1 : N - 2$ 
 $U_i = \frac{1}{f_{i+1}} \begin{bmatrix} u_{i+1} & -f_i \\ f_i & u_{i+1} \end{bmatrix};$ 
 $\tilde{U}_i = \text{diag}(I_{i-1}, U_i, I_{N-i-1});$ 
 $A^{(i)} = \tilde{U}_i A^{(i-1)} \tilde{U}_i^T;$ 
end

```

LEMMA 2.1. *The matrices  $A^{(i)} = \left(a_{k,j}^{(i)}\right)_{k,j=1}^N$  generated by Algorithm 1 are similar to  $A$  and have the following structure:*

$$(5) \quad A^{(i)} = \left[ \begin{array}{cccc|ccc} a_{1,1}^{(i)} & \cdots & a_{1,i}^{(i)} & a_{1,i+1}^{(i)} & 0 & \cdots & 0 \\ \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ a_{i,1}^{(i)} & \cdots & a_{i,i}^{(i)} & a_{i,i+1}^{(i)} & 0 & \cdots & 0 \\ \hline a_{i+1,1}^{(i)} & \cdots & a_{i+1,i}^{(i)} & a_{i+1,i+1}^{(i)} & a_{i+1,i+2}^{(i)} & \cdots & a_{i+1,N}^{(i)} \\ 0 & \cdots & 0 & a_{i+2,i+1}^{(i)} & a_{i+2,i+2}^{(i)} & \cdots & a_{i+2,N}^{(i)} \\ \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & \cdots & 0 & a_{N,i+1}^{(i)} & a_{N,i+2}^{(i)} & \cdots & a_{N,N}^{(i)} \end{array} \right].$$

Moreover, for  $0 \leq i \leq N-2$  define

$$\tilde{A}^{(i)} = \left[ \begin{array}{cccc} a_{1,1}^{(i)} & \cdots & a_{1,i}^{(i)} & a_{1,i+1}^{(i)} \\ \vdots & \cdots & \vdots & \vdots \\ a_{i,1}^{(i)} & \cdots & a_{i,i}^{(i)} & a_{i,i+1}^{(i)} \\ a_{i+1,1}^{(i)} & \cdots & a_{i+1,i}^{(i)} & a_{i+1,i+1}^{(i)} \end{array} \right]$$

and

$$\hat{A}^{(i)} = \left[ \begin{array}{cccc} a_{i+1,i+1}^{(i)} & a_{i+1,i+2}^{(i)} & \cdots & a_{i+1,N}^{(i)} \\ a_{i+2,i+1}^{(i)} & a_{i+2,i+2}^{(i)} & \cdots & a_{i+2,N}^{(i)} \\ \vdots & \vdots & \cdots & \vdots \\ a_{N,i+1}^{(i)} & a_{N,i+2}^{(i)} & \cdots & a_{N,N}^{(i)} \end{array} \right].$$

Thus,  $\tilde{A}^{(i)}$  and  $\hat{A}^{(i)}$  are DSS matrices. Indeed, let  $\mathbf{u}^{(0)}$ ,  $\mathbf{v}^{(0)}$  and  $\mathbf{x}^{(0)}$  redefine the vectors  $\mathbf{u}$ ,  $\mathbf{v}$  and  $\mathbf{x}$  in (2), respectively; since the value of  $v_1$  is arbitrary, we assume that  $v_1 = v_1^{(0)} \neq 0$ . Let  $\mathbf{u}^{(i)} = [u_1^{(i)}, \dots, u_N^{(i)}]^T$ ,  $\mathbf{v}^{(i)} = [v_1^{(i)}, \dots, v_N^{(i)}]^T$  and  $\mathbf{x}^{(i)} = [x_1^{(i)}, \dots, x_N^{(i)}]^T$ , for  $1 \leq i \leq N-2$ , be as follows:

$$(6) \quad j \notin \{i, i+1\} \implies u_j^{(i)} = u_j^{(i-1)}, \quad v_j^{(i)} = v_j^{(i-1)}, \quad x_j^{(i)} = x_j^{(i-1)}.$$

Moreover,

$$(7) \quad u_{i+1}^{(i)} = f_{i+1}$$

$$(8) \quad v_i^{(i)} = v_i^{(i-1)} u_{i+1} / f_{i+1}$$

$$(9) \quad v_{i+1}^{(i)} = v_i^{(i-1)} f_i / f_{i+1}$$

where  $f_1, \dots, f_{N-1}$  are given in (4). Then,  $v_{i+1}^{(i)} \neq 0$ . Furthermore, define  $x_i^{(i)}$ ,  $x_{i+1}^{(i)}$  and  $u_i^{(i)}$  from the previously defined quantities so that the matrix

$$\begin{bmatrix} x_i^{(i)} & u_i^{(i)} v_{i+1}^{(i)} \\ u_i^{(i)} v_{i+1}^{(i)} & x_{i+1}^{(i)} \end{bmatrix}$$

is equal to

$$\frac{1}{f_{i+1}^2} \begin{bmatrix} u_{i+1} & -f_i \\ f_i & u_{i+1} \end{bmatrix} \begin{bmatrix} x_i^{(i-1)} & u_i^{(i-1)} v_{i+1}^{(i-1)} \\ u_i^{(i-1)} v_{i+1}^{(i-1)} & x_{i+1}^{(i-1)} \end{bmatrix} \begin{bmatrix} u_{i+1} & f_i \\ -f_i & u_{i+1} \end{bmatrix}.$$

Then,

$$(10) \quad \tilde{A}^{(i)} = \mathcal{DSS}(\mathbf{u}^{(i)}(1:i+1), \mathbf{v}^{(i)}(1:i+1), \mathbf{x}^{(i)}(1:i+1))$$

$$(11) \quad \hat{A}^{(i)} = \mathcal{DSS}(\mathbf{u}^{(i)}(i+1:N), \mathbf{v}^{(i)}(i+1:N), \mathbf{x}^{(i)}(i+1:N)).$$

PROOF. First of all, observe that the transformation  $A^{(i)} = \tilde{U}_i A^{(i-1)} \tilde{U}_i^T$  preserves symmetry. Actually, this transformation is a similarity. Indeed, taking the definition of  $f_j$  in (4) into account, we get

$$U_i^T U_i = \frac{1}{f_{i+1}^2} \begin{bmatrix} u_{i+1} & f_i \\ -f_i & u_{i+1} \end{bmatrix} \begin{bmatrix} u_{i+1} & -f_i \\ f_i & u_{i+1} \end{bmatrix} = \frac{1}{f_{i+1}^2} \begin{bmatrix} f_{i+1}^2 & 0 \\ 0 & f_{i+1}^2 \end{bmatrix} = I_2.$$

Hence the matrices  $\tilde{U}_i$ , for  $i = 1, \dots, N-2$ , are Givens rotations. Since  $A^{(i)}$  differs from  $A^{(i-1)}$  only in the  $i$ -th and  $(i+1)$ -th rows and columns, to assess (10) and (11) we must check only the form of the affected rows and columns.

The lemma is proven by finite induction. Let's prove it for  $i = 1$ . By construction, we obtain trivially the identity

$$\frac{1}{f_2^2} \begin{bmatrix} u_2 & -f_1 \\ f_1 & u_2 \end{bmatrix} \begin{bmatrix} x_1^{(0)} & u_1^{(0)} v_2^{(0)} \\ u_1^{(0)} v_2^{(0)} & x_2^{(0)} \end{bmatrix} \begin{bmatrix} u_2 & f_1 \\ -f_1 & u_2 \end{bmatrix} = \begin{bmatrix} x_1^{(1)} & u_1^{(1)} v_2^{(1)} \\ u_1^{(1)} v_2^{(1)} & x_2^{(1)} \end{bmatrix}.$$

Indeed, the hypotheses  $v_1^{(0)} \neq 0$  and  $f_1 = u_1 \neq 0$  ensure that the value of  $v_2^{(1)}$  computed in (9) is different from zero. Hence a suitable value for  $u_i^{(i)}$  can be found, and we have (10). From equation (7) we have, for  $j = 3, \dots, N$ ,

$$\begin{bmatrix} a_{1,j}^{(1)} \\ a_{2,j}^{(1)} \end{bmatrix} = \frac{1}{f_2} \begin{bmatrix} u_2 & -f_1 \\ f_1 & u_2 \end{bmatrix} \begin{bmatrix} a_{1,j}^{(0)} \\ a_{2,j}^{(0)} \end{bmatrix} = \frac{1}{f_2} \begin{bmatrix} u_2 u_1 v_j - f_1 u_2 v_j \\ f_1 u_1 v_j + u_2 u_2 v_j \end{bmatrix} = \begin{bmatrix} 0 \\ f_2 v_j \end{bmatrix} = \begin{bmatrix} 0 \\ u_2^{(1)} v_j^{(1)} \end{bmatrix}.$$

This proves (11) and also that  $A^{(1)}$  has the form (5).

Now suppose the claim is true for  $i-1$ , with  $0 < i < N-1$ . We prove it for  $i$ . We have:

$$\begin{aligned} A^{(i)} &= \tilde{U}_i A^{(i-1)} \tilde{U}_i^T \\ &= \tilde{U}_i \left[ \begin{array}{cccc|ccc} a_{1,1}^{(i-1)} & \cdots & a_{1,i-1}^{(i-1)} & a_{1,i}^{(i-1)} & 0 & \cdots & 0 \\ \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ a_{i-1,1}^{(i-1)} & \cdots & a_{i-1,i-1}^{(i-1)} & a_{i-1,i}^{(i-1)} & 0 & \cdots & 0 \\ a_{i,1}^{(i-1)} & \cdots & a_{i,i-1}^{(i-1)} & a_{i,i}^{(i-1)} & a_{i,i+1}^{(i-1)} & \cdots & a_{i,N}^{(i-1)} \\ \hline 0 & \cdots & 0 & a_{i+1,i}^{(i-1)} & a_{i+1,i+1}^{(i-1)} & \cdots & a_{i+1,N}^{(i-1)} \\ \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & \cdots & 0 & a_{N,i}^{(i-1)} & a_{N,i+1}^{(i-1)} & \cdots & a_{N,N}^{(i-1)} \end{array} \right] \tilde{U}_i^T. \end{aligned}$$

From (9) we obtain  $v_{i+1}^{(i)} \neq 0$  since  $v_i^{(i-1)} \neq 0$  by inductive hypothesis and  $f_i \neq 0$ . Moreover, for  $j = 1, \dots, i-1$  we have  $a_{i,j}^{(i-1)} = u_j^{(i-1)} v_i^{(i-1)}$ . From (8) we obtain

$$\begin{aligned} \begin{bmatrix} a_{i,j}^{(i)} \\ a_{i+1,j}^{(i)} \end{bmatrix} &= \frac{1}{f_{i+1}} \begin{bmatrix} u_{i+1} & -f_i \\ f_i & u_{i+1} \end{bmatrix} \begin{bmatrix} a_{i,j}^{(i-1)} \\ 0 \end{bmatrix} \\ &= \frac{1}{f_{i+1}} \begin{bmatrix} u_{i+1} u_j^{(i-1)} v_i^{(i-1)} \\ f_i u_j^{(i-1)} v_i^{(i-1)} \end{bmatrix} = \begin{bmatrix} v_i^{(i)} u_j^{(i)} \\ v_{i+1}^{(i)} u_j^{(i)} \end{bmatrix}. \end{aligned}$$

This confirms (10). On the other hand, for  $j = i + 2, \dots, N$ , we have

$$\begin{aligned} \begin{bmatrix} a_{i,j}^{(i)} \\ a_{i+1,j}^{(i)} \end{bmatrix} &= \frac{1}{f_{i+1}} \begin{bmatrix} u_{i+1} & -f_i \\ f_i & u_{i+1} \end{bmatrix} \begin{bmatrix} a_{i,j}^{(i-1)} \\ a_{i+1,j}^{(i-1)} \end{bmatrix} \\ &= \frac{1}{f_{i+1}} \begin{bmatrix} u_{i+1} & -f_i \\ f_i & u_{i+1} \end{bmatrix} \begin{bmatrix} f_i v_j \\ u_{i+1} v_j \end{bmatrix} \\ &= \frac{v_j}{f_{i+1}} \begin{bmatrix} u_{i+1} f_i - f_i u_{i+1} \\ f_i f_i + u_{i+1} u_{i+1} \end{bmatrix} = \begin{bmatrix} 0 \\ v_j f_{i+1} \end{bmatrix}. \end{aligned}$$

Hence we have (11); moreover, we see that  $A^{(i)}$  has the form (5), and the proof is complete.  $\square$

The matrix  $A^{(N-2)}$ , computed by Algorithm 1, is a  $2 \times 2$  block matrix,

$$A^{(N-2)} = \left[ \begin{array}{c|c} \tilde{A}^{(N-2)} & \begin{matrix} 0 \\ \vdots \\ 0 \end{matrix} \\ \hline \begin{matrix} 0 & \cdots & 0 & a_{N,N-1}^{(N-2)} \end{matrix} & \begin{matrix} a_{N-1,N}^{(N-2)} \\ a_{N,N}^{(N-2)} \end{matrix} \end{array} \right],$$

where  $A_1 \equiv \tilde{A}^{(N-2)}$  is DSS. We remark that the last row and column of  $A^{(N-2)}$  are already in a tridiagonal form. Moreover, Algorithm 1 leaves the  $(N, N)$  entry of  $A$  unaltered, that is,  $a_{N,N}^{(N-2)} = x_N$ . Hence, if we apply the first  $N - 3$  steps of the preceding algorithm to the matrix  $A^{(N-2)}$  above, its last row and column are left unchanged, and we can consider the recursive application of Algorithm 1 to the submatrix  $A_1$ . Then, the complete reduction of  $A_0 \equiv A$  into tridiagonal form can be accomplished by applying Algorithm 1 to the recursively generated leading submatrices  $A_i$ , for  $i = 1, \dots, N - 3$ .

At each step of Algorithm 1, underflows and overflows can occur in the computation of (4). This drawback can be circumvented by dividing, at the beginning of the  $i$ -th step, the subvectors  $\mathbf{u}^{(i)}(1 : N - i)$  and  $\mathbf{x}^{(i)}(1 : N - i + 1)$  by  $u_{max}^{(i)} = \|\mathbf{u}^{(i)}(1 : N - i)\|_\infty$  and multiplying, at the end of the same step, the currently computed elements of the tridiagonal matrix by  $u_{max}^{(i)}$ .

The complete algorithm, written in a `matlab`-like style, is summarized in Table 1. The input of the function `tridDSS1` are the vectors  $\mathbf{u}, \mathbf{v}, \mathbf{x}$  and the output are the main diagonal  $\mathbf{d}$  and the codiagonal  $\mathbf{b}$  of the symmetric tridiagonal matrix (3). The number of flops needed is  $15N^2 + O(N)$ .

### 3. Two-way algorithm

The algorithm described in § 2 can be easily adapted to reduce a DSS matrix into a similar tridiagonal one starting from the south-east corner of  $A$ . In the latter case, define

$$(12) \quad g_N = v_N, \quad g_j = \sqrt{\sum_{k=j}^N v_k^2}, \quad j = N - 1, N - 2, \dots, 2.$$

The algorithm described in § 2 becomes

```

function[d,b] = tridDSS1 (u,v,x);
N = length(x);
smax = 1;
for j = N - 2 : -1 : 1,
    umax = max(u(1:j+1));
    u(1:j+1) = u(1:j+1)/umax;
    xj+2 = xj+2smax;
    x(1:j+1) = x(1:j+1)/umax;
    smax = smaxumax;
    n1 = u1^2 + u2^2;
    f2 = sqrt(n1);
    f1 = u1;
    c = u2/f2;   c2 = c^2;
    s = f1/f2;   s2 = s^2;
    for i = 1:j,
        t1 = ui+1(xi - xi+1);
        t2 = ui vi+1;
        t3 = xi;
        t4 = 2cst2;
        di = c2xi + s2xi+1 - t4;
        di+1 = s2t3 + c2xi+1 + t4;
        ui = st1 + t2(c - s)(ui+1 - f2);
        ui+1 = f2;
        f1 = f2;
        if i ≠ j,
            n1 = n1 + u(i+2)^2;
            f2 = sqrt(n1);
            c = ui+2/f2;   c2 = c^2;
            s = f1/f2;   s2 = s^2;
            vi+1 = ui+2/(f1f2);
        else
            vi+1 = 1/f2;
        end
    end
    b(j+1) = f1v(j+2)smax;
end
b1 = u1v2smax;
d1 = d1smax;
d2 = d2smax;

```

TABLE 1. matlab-like function **tridDSS1****Algorithm 2**

$$B^{(0)} = A;$$

$$N = \max(\text{size}(B^{(0)}));$$

**for**  $i = N : -1 : 3$

$$V_i = \frac{1}{g_{i-1}} \begin{bmatrix} v_{i-1} & g_i \\ -g_i & v_{i-1} \end{bmatrix};$$

$$\tilde{V}_i = \text{diag}(I_{i-2}, V_i, I_{N-i});$$

$$B^{(N-i+1)} = \tilde{V}_i B^{(N-i)} \tilde{V}_i^T;$$

**end**

Also in this case, the matrices  $B^{(j)}$ , for  $j = 0, \dots, N-2$ , can be proven to be similar to  $A$ . Moreover, the matrix  $B^{(N-2)}$  is a  $2 \times 2$  block matrix,

$$B^{(N-2)} = \left[ \begin{array}{c|ccccc} a_{1,1}^{(N-2)} & a_{1,2}^{(N-2)} & 0 & \cdots & 0 \\ \hline a_{2,1}^{(N-2)} & 0 & B_1 & & \\ \vdots & & & & \\ 0 & & & & \end{array} \right]$$

where  $B_1$  is DSS. Hence the reduction of  $A$  into tridiagonal form can be accomplished by recursively applying Algorithm 2 to the matrices  $B_i$ , for  $i = 1, \dots, N-3$ , analogously to the argument presented in the preceding section. We refer to this algorithm as **tridDSS2**<sup>1</sup>. Of course, this algorithm has the same computational complexity as the algorithm described in Table 1. We observe that the  $(1, 1)$ -entry in the matrices  $B_i^{(0)}$ ,  $i = 1, \dots, N-2$ , is not modified by **tridDSS2**.

Combining both these algorithms we can consider a procedure for the reduction of a DSS matrix into a similar tridiagonal one that halves the computational complexity. Indeed, if we apply  $M = \lfloor \frac{N}{2} \rfloor + 2$  steps of Algorithm 1 to  $A$ , the resulting matrix  $A^{(M)}$  is

$$A^{(M)} = \left[ \begin{array}{cccc|ccc} a_{1,1}^{(M)} & \cdots & a_{1,M}^{(M)} & a_{1,M+1}^{(M)} & 0 & \cdots & 0 \\ \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ a_{M,1}^{(M)} & \cdots & a_{M,M}^{(M)} & a_{M,M+1}^{(M)} & 0 & \cdots & 0 \\ \hline a_{M+1,1}^{(M)} & \cdots & a_{M+1,M}^{(M)} & a_{M+1,M+1}^{(M)} & a_{M+1,M+2}^{(M)} & \cdots & a_{M+1,N}^{(M)} \\ 0 & \cdots & 0 & a_{M+2,M+1}^{(M)} & a_{M+2,M+2}^{(M)} & \cdots & a_{M+2,N}^{(M)} \\ \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & \cdots & 0 & a_{N,M+1}^{(M)} & a_{N,M+2}^{(M)} & \cdots & a_{N,N}^{(M)} \end{array} \right].$$

Define

$$\tilde{A} = \left[ \begin{array}{cccc} a_{1,1}^{(M)} & \cdots & a_{1,M}^{(M)} & a_{1,M+1}^{(M)} \\ \vdots & \cdots & \vdots & \vdots \\ a_{M,1}^{(M)} & \cdots & a_{M,M}^{(M)} & a_{M,M+1}^{(M)} \\ a_{M+1,1}^{(M)} & \cdots & a_{M+1,M}^{(M)} & a_{M+1,M+1}^{(M)} \end{array} \right]$$

<sup>1</sup>We omit the **matlab**-like code for the sake of brevity. The interested reader can obtain all the **matlab** codes described in the paper from the authors upon request.

and

$$\hat{A} = \begin{bmatrix} a_{M+1,M+1}^{(M)} & a_{M+1,M+2}^{(M)} & \cdots & a_{M+1,N}^{(M)} \\ a_{M+2,M+1}^{(M)} & a_{M+2,M+2}^{(M)} & \cdots & a_{M+2,N}^{(M)} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N,M+1}^{(M)} & a_{N,M+2}^{(M)} & \cdots & a_{N,N}^{(M)} \end{bmatrix}.$$

From Lemma 2.1, both  $\tilde{A}$  and  $\hat{A}$  are DSS matrices. Hence, the reduction of  $A$  into a similar tridiagonal matrix is accomplished by applying `tridDSS1` and `tridDSS2` to  $\tilde{A}$  and  $\hat{A}$ , respectively. This is possible, since the entry in the lower right corner of  $\tilde{A}$  is left unchanged by `tridDSS1`, as well as the entry in the upper left corner of  $\hat{A}$  is left unchanged by `tridDSS2`. The corresponding function `tridDSS3` has a computational cost  $7.5N^2 + O(N)$  flops.

We observe that the techniques just described can be easily adapted to the wider class of symmetric band plus semiseparable matrices considered in [4]. Indeed, if  $A$  can be written as the sum of a symmetric band matrix, having bandwidth  $2r - 1$ , and a symmetric semiseparable one,

$$A = \begin{bmatrix} e_{1,1} & e_{1,2} & \cdots & e_{1,r} & u_1 v_{r+1} & \cdots & u_1 v_N \\ e_{2,1} & e_{2,2} & \ddots & \ddots & e_{2,r+1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & u_{N-r} v_N \\ e_{r,1} & e_{r,2} & \ddots & \ddots & \ddots & \ddots & e_{N-r+1,N} \\ u_1 v_{r+1} & e_{r+1,2} & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & e_{N-1,N-1} & e_{N-1,N} \\ u_1 v_N & \cdots & u_{N-r} v_N & e_{N,N-r+1} & \cdots & e_{N,N-1} & e_{N,N} \end{bmatrix},$$

then the reduction into a similar tridiagonal matrix can be accomplished in two steps,

- (1) reduction of  $A$  into a similar band matrix with bandwidth  $2r + 1$ ,
- (2) reduction of  $A$  into a similar tridiagonal matrix.

Using the techniques described in this paper, the first step can be accomplished in  $O(r(N-r)^2)$  flops. The second step above can be implemented in a fast and stable way by means of classical techniques [15], requiring  $O(r(N-r)^2)$  flops.

#### 4. Bidiagonalizing diagonal plus unsymmetric semiseparable matrices.

In this section we consider an algorithm to reduce the diagonal plus unsymmetric semiseparable (DSSU) matrix

$$(13) \quad R = \begin{bmatrix} x_1 & u_1 v_2 & u_1 v_3 & \cdots & u_1 v_{N-1} & u_1 v_N \\ z_1 w_2 & x_2 & u_2 v_3 & \cdots & u_2 v_{N-1} & u_2 v_N \\ z_1 w_3 & z_2 w_3 & \ddots & \cdots & \cdots & \cdots \\ z_1 w_4 & z_2 w_4 & \vdots & \ddots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & x_{N-1} & u_{N-1} v_N \\ z_1 w_N & z_2 w_N & \vdots & \vdots & z_{N-1} w_N & x_N \end{bmatrix}.$$

into bidiagonal form by means of orthogonal transformations. Such reduction occurs as the first step toward the computation of the SVD of  $R$ , see e.g., [10].

In what follows, we will say that the above matrix  $R$  is *generated* by the vectors  $\mathbf{u}$ ,  $\mathbf{v}$ ,  $\mathbf{z}$ ,  $\mathbf{w}$  and  $\mathbf{x}$ . The algorithm is divided into three parts:

- (1) transformation of  $R$  into an unsymmetric tridiagonal matrix;
- (2) transformation of the unsymmetric tridiagonal matrix into an upper tridiagonal one;
- (3) transformation of the upper tridiagonal matrix into an upper bidiagonal one.

Parts 2 and 3 can be accomplished in the way described in [15, 12]. Concerning Part 1, we observe that the matrix (13) can be transformed into an unsymmetric tridiagonal matrix in a way similar to that one considered in § 2 to reduce a diagonal plus a symmetric semiseparable matrix into a symmetric tridiagonal one, proceeding in  $N - 2$  steps as follows. In the first step, we use some Givens rotations to bring the matrix  $R$  in the following form:

$$\left[ \begin{array}{ccc|c} & & & 0 \\ R_1 & & & \vdots \\ & & & 0 \\ \hline 0 & \cdots & 0 & c_{N-1} \\ & & & d_N \end{array} \right],$$

where  $R_1$  is an unsymmetric DSS. Then, the same procedure is applied recursively starting from  $R_1$ , until the leading block is a square matrix of order two, ending up with an unsymmetric tridiagonal matrix,

$$(14) \quad \left[ \begin{array}{ccc} d_1 & b_1 & O \\ c_1 & d_2 & \ddots \\ \ddots & \ddots & b_{N-1} \\ O & c_{N-1} & d_N \end{array} \right].$$

In what follows, we will suppose  $u_1, z_1 \neq 0$ . If this is not the case, minor variations are needed only in the very first step of the subsequent algorithm. Define the scalars  $f_j$  and  $h_j$ ,  $j = 1, \dots, N - 1$ , as follows:

$$(15) \quad f_1 = u_1, \quad h_1 = z_1, \quad f_j = \sqrt{\sum_{k=1}^j u_k^2}, \quad h_j = \sqrt{\sum_{k=1}^j z_k^2}, \quad j = 2, \dots, N - 1.$$

We describe the generic step of our algorithm as follows:

**Algorithm 2**

$$R^{(0)} = R;$$

$$N = \max(\text{size}(R^{(0)})) ;$$

**for**  $i = 1 : N - 2$ ,

$$U_i = \frac{1}{f_{i+1}} \begin{bmatrix} u_{i+1} & -f_i \\ f_i & u_{i+1} \end{bmatrix}; \quad V_i = \frac{1}{h_{i+1}} \begin{bmatrix} z_{i+1} & -h_i \\ h_i & z_{i+1} \end{bmatrix};$$

$$\tilde{U}_i = \text{diag}(I_{i-1}, U_i, I_{N-i-1}); \quad \tilde{V}_i = \text{diag}(I_{i-2}, V_i, I_{N-i}) ;$$

$$R^{(i)} = \tilde{U}_i R^{(i-1)} \tilde{V}_i^T$$

end

The following lemma mirrors the analogous result in Section 2, and can be proven by the same technique used in proving Lemma 4.1. For the sake of brevity we omit the details of its proof.

LEMMA 4.1. *The matrices  $R^{(i)} = \left(r_{k,j}^{(i)}\right)_{k,j=1}^N$ ,  $i = 1, \dots, N-2$ , generated by Algorithm 2, have the same singular values of  $R$  and have the following structure,*

$$R^{(i)} = \begin{bmatrix} r_{1,1}^{(i)} & \cdots & r_{1,i}^{(i)} & r_{1,i+1}^{(i)} & 0 & \cdots & 0 \\ \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ r_{i,1}^{(i)} & \cdots & r_{i,i}^{(i)} & r_{i,i+1}^{(i)} & 0 & \cdots & 0 \\ \hline r_{i+1,1}^{(i)} & \cdots & r_{i+1,i}^{(i)} & r_{i+1,i+1}^{(i)} & r_{i+1,i+2}^{(i)} & \cdots & r_{i+1,N}^{(i)} \\ 0 & \cdots & 0 & r_{i+2,i+1}^{(i)} & r_{i+2,i+2}^{(i)} & \cdots & r_{i+2,N}^{(i)} \\ \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & \cdots & 0 & r_{N,i+1}^{(i)} & r_{N,i+2}^{(i)} & \cdots & r_{N,N}^{(i)} \end{bmatrix} .$$

Moreover, for  $0 \leq i \leq N-2$  define

$$\tilde{R}^{(i)} = \begin{bmatrix} r_{1,1}^{(i)} & \cdots & r_{1,i}^{(i)} & r_{1,i+1}^{(i)} \\ \vdots & \cdots & \vdots & \vdots \\ r_{i,1}^{(i)} & \cdots & r_{i,i}^{(i)} & r_{i,i+1}^{(i)} \\ r_{i+1,1}^{(i)} & \cdots & r_{i+1,i}^{(i)} & r_{i+1,i+1}^{(i)} \end{bmatrix}$$

and

$$\hat{R}^{(i)} = \begin{bmatrix} r_{i+1,i+1}^{(i)} & r_{i+1,i+2}^{(i)} & \cdots & r_{i+1,N}^{(i)} \\ r_{i+2,i+1}^{(i)} & r_{i+2,i+2}^{(i)} & \cdots & r_{i+2,N}^{(i)} \\ \vdots & \vdots & \cdots & \vdots \\ r_{N,i+1}^{(i)} & r_{N,i+2}^{(i)} & \cdots & r_{N,N}^{(i)} \end{bmatrix} .$$

Then the matrices  $\tilde{R}^{(i)}$  and  $\hat{R}^{(i)}$ , are DSSU matrices. Indeed, let  $\mathbf{u}^{(0)}, \mathbf{v}^{(0)}, \mathbf{z}^{(0)}$ ,  $\mathbf{w}^{(0)}$ , and  $\mathbf{x}^{(0)}$  redefine the vectors  $\mathbf{u}, \mathbf{v}, \mathbf{z}, \mathbf{w}$  and  $\mathbf{x}$  that generate the matrix  $R$  in (13). Let  $\mathbf{u}^{(i)} = [u_1^{(i)}, \dots, u_N^{(i)}]^T$ ,  $\mathbf{v}^{(i)} = [v_1^{(i)}, \dots, v_N^{(i)}]^T$ ,  $\mathbf{z}^{(i)} = [z_1^{(i)}, \dots, z_N^{(i)}]^T$ ,  $\mathbf{w}^{(i)} = [w_1^{(i)}, \dots, w_N^{(i)}]^T$  and  $\mathbf{x}^{(i)} = [x_1^{(i)}, \dots, x_N^{(i)}]^T$ , for  $1 \leq i \leq N-2$ , be as follows:

$$j \notin \{i, i+1\} \implies \begin{cases} u_j^{(i)} = u_j^{(i-1)}, v_j^{(i)} = v_j^{(i-1)}, z_j^{(i)} = z_j^{(i-1)}, \\ w_j^{(i)} = w_j^{(i-1)}, x_j^{(i)} = x_j^{(i-1)}. \end{cases}$$

Moreover,

$$\begin{aligned} u_{i+1}^{(i)} &= f_{i+1} & z_{i+1}^{(i)} &= h_{i+1} \\ v_i^{(i)} &= v_i^{(i-1)} u_{i+1}/h_{i+1} & w_i^{(i)} &= w_i^{(i-1)} z_{i+1}/f_{i+1} \\ v_{i+1}^{(i)} &= v_i^{(i-1)} h_i/h_{i+1} & w_{i+1}^{(i)} &= w_i^{(i-1)} f_i/f_{i+1} \end{aligned}$$

where  $f_k$ ,  $h_k$ ,  $k = 1, \dots, N - 1$ , are given in (15). Then,  $v_{i+1}^{(i)} \neq 0$  and  $w_{i+1}^{(i)} \neq 0$ . Finally, define  $x_i^{(i)}$ ,  $x_{i+1}^{(i)}$  and  $u_i^{(i)}$  from the previously defined quantities so that the matrix

$$\begin{bmatrix} x_i^{(i)} & u_i^{(i)} v_{i+1}^{(i)} \\ z_i^{(i)} w_{i+1}^{(i)} & x_{i+1}^{(i)} \end{bmatrix}$$

is equal to

$$\frac{1}{f_{i+1} h_{j+1}} \begin{bmatrix} u_{i+1} & -f_i \\ f_i & u_{i+1} \end{bmatrix} \begin{bmatrix} x_i^{(i-1)} & u_i^{(i-1)} v_{i+1}^{(i-1)} \\ z_i^{(i-1)} w_{i+1}^{(i-1)} & x_{i+1}^{(i-1)} \end{bmatrix} \begin{bmatrix} z_{i+1} & h_i \\ -h_i & z_{i+1} \end{bmatrix}.$$

Then, the matrix  $\tilde{R}^{(i)}$  is the DSSU matrix generated by the subvectors of  $\mathbf{u}^{(i)}$ ,  $\mathbf{v}^{(i)}$ ,  $\mathbf{z}^{(i)}$ ,  $\mathbf{w}^{(i)}$  and  $\mathbf{x}^{(i)}$  made up by their first  $i + 1$  entries; and the matrix  $\hat{R}^{(i)}$  is the DSSU matrix generated by the subvectors made up by the last  $N - i$  entries of the same vectors.

## 5. Numerical results

In this section we compare the accuracy of the proposed algorithms in computing the eigenvalues of diagonal plus semiseparable matrices. In particular we consider the following DSS matrices,

$$(16) \quad A_N = \mathcal{DSS}(\mathbf{u}_N, \mathbf{v}_N, \mathbf{x}_N), \quad N = 40, 80, \dots, 640, 680,$$

where

$$\begin{aligned} \mathbf{u}_N &= [1, 2, \dots, N - 1, N]^T \\ \mathbf{v}_N &= [N, N - 1, \dots, 2, 1]^T \\ \mathbf{x}_N &= [N, 2(N - 1), 3(N - 2), \dots, N]. \end{aligned}$$

Such matrices arise in the theory of statistics in the variance–covariance matrices of ordered observations of random samples of size  $N$  from rectangular distributions [11]. We observe that the matrices  $A_N$  are the inverses of the symmetric tridiagonal Toeplitz matrices [11]

$$\begin{bmatrix} 2(N+1)^{-1} & -(N+1)^{-1} & & O \\ -(N+1)^{-1} & 2(N+1)^{-1} & \ddots & \\ & \ddots & \ddots & -(N+1)^{-1} \\ O & & -(N+1)^{-1} & 2(N+1)^{-1} \end{bmatrix}_{N \times N},$$

whose eigenvalues are

$$\lambda_{j,N} = \frac{2}{N+1} \left( 1 - \cos \frac{j\pi}{N+1} \right), \quad j = 1, \dots, N.$$

All the computations were performed in **matlab** 6.0 on a Sun Workstation. The results are reported in Table 2. The order of the considered matrices can be found in the first column. The *relative residuals*

$$res^{(QR)} = \max_{i \in \{1, \dots, N\}} \left| \frac{\lambda_{i,N} - \lambda_{i,N}^{(QR)}}{\lambda_{i,N}} \right|, \quad res^{(TQR)} = \max_{i \in \{1, \dots, N\}} \left| \frac{\lambda_{i,N} - \lambda_{i,N}^{(TQR)}}{\lambda_{i,N}} \right|,$$

N	$res^{(QR)}$	$res^{(TQR)}$
40	8.8521e-15	3.7696e-15
80	3.3912e-14	3.5128e-14
120	1.0830e-13	1.0944e-13
160	3.8722e-13	3.8763e-13
200	5.5689e-13	5.5533e-13
240	2.5757e-13	2.5822e-13
280	1.1558e-12	1.1577e-12
320	3.5932e-13	3.5473e-13
360	1.8690e-12	1.8709e-12
400	3.5245e-12	3.5251e-12
440	2.5001e-12	2.5023e-12
480	4.6006e-12	4.6005e-12
520	3.3466e-12	3.3448e-12
560	1.7265e-12	1.7259e-12
600	2.0731e-12	2.0724e-12
640	5.1948e-12	5.1949e-12
680	3.9113e-12	3.9113e-12

TABLE 2. *Relative residual for the eigenvalue computed by the Matlab routine eig and by the same routine after the transformation of the DSS matrices (16) into similar tridiagonal ones by means of the routine tridDSS3*

where  $\lambda_{i,N}^{(QR)}$  and  $\lambda_{i,N}^{(TQR)}$  are the eigenvalues computed by the **matlab** function **eig** and by the same function after the transformation of the matrices (16) into similar tridiagonal ones by means of the routine **tridDSS3** described in § 3, are reported in the second and the third column, respectively.

We can observe that the eigenvalues computed by first reducing the considered symmetric DSS matrices into tridiagonal ones by means of **tridDSS3** are as accurate as those computed by first reducing the considered symmetric DSS matrices into tridiagonal ones by means of Householder transformations [10].

## 6. Conclusions

A new algorithm to reduce a diagonal plus a symmetric semiseparable matrix to a tridiagonal one have been presented in § 2. Moreover, a two-way algorithm to reduce a diagonal plus a symmetric semiseparable matrix to a tridiagonal one has been presented in § 3. The required number of flops for the two-way algorithm is half of that one described in § 2. This algorithm can be easily generalized to a banded plus semiseparable matrix whose off-diagonal part has rank bigger than 1 [4]. In § 4, a fast algorithm to reduce a diagonal plus an unsymmetric semiseparable matrix to a bidiagonal one has been presented. This algorithm is divided into three steps requiring a computational cost of order  $N^2$ . Finally, in § 5, the numerical experiments confirm that the proposed algorithms are as accurate as the standard algorithms for solving the same problems.

## References

- [1] A. Abdallah and Y. Hu, *Parallel VLSI computing array implementation for signal subspace updating algorithm*, IEEE Trans. Acoust., Speech and Signal Processing, ASSP-37, pp. 742–748, 1989.
- [2] Å. Björck , *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996.
- [3] S. Chandrasekaran and M. Gu, *A Divide-and-Conquer algorithm for the eigendecomposition of symmetric block-diagonal plus semiseparable matrices*, submitted for publication.
- [4] S. Chandrasekaran and M. Gu, *Fast and stable eigendecomposition of symmetric banded plus semiseparable matrices*, Linear Algebra Appl., 313, pp. 107–114, 2000.
- [5] P. M. Dewilde and A. J. van der Veen, *Time-varying systems and computations*, Kluwer, 1998.
- [6] Y. Eidelman and I. Gohberg, *Inversion formulas and linear complexity algorithm for diagonal plus semiseparable matrices*, Comput. Math. Appl., 33, pp. 69–79, 1997.
- [7] Y. Eidelman and I. Gohberg, *On a new class of structured matrices*, Integral Equations Operator Theory, 34, pp. 293–324, 1999.
- [8] Y. Eidelman and I. Gohberg, *A modification of the dewilde van der veen method for inversion of finite structured matrices*, Linear Algebra Appl., 343–344, pp. 419–450, April 2002.
- [9] L. Eldén, *Algorithms for the regularization of ill-conditioned least squares problems*, BIT, 17, pp. 134–145, 1979.
- [10] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Third ed., The John Hopkins University Press, Baltimore, MD, 1996.
- [11] F.A. Graybill, *Matrices with Applications in Statistics*, Second ed., The Wadsworth Statistics/Probability Series, Belmont, California, 1983.
- [12] N. Mastronardi, S. Chandrasekaran and S. Van Huffel, *Fast and Stable algorithms for reducing diagonal plus semiseparable matrices to tridiagonal and bidiagonal form*, BIT, 41, pp. 149–157, 2001.
- [13] N. Mastronardi, S. Chandrasekaran, and S. Van Huffel, *Fast and stable two-way algorithm for diagonal plus semi-separable systems of linear equations*, Numer. Linear Algebra Appl., 8(1), pp. 7–12, 2001.
- [14] B. N. Parlett, *The Symmetric Eigenvalue Problem*, Prentice Hall, Englewood Cliffs, NJ, 1980.
- [15] H. R. Schwarz, *Tridiagonalization of a symmetric banded matrix*, Numer. Math., 12, pp. 231–241, 1968.
- [16] S. Van Huffel and H. Park, *Efficient reduction algorithms for bordered matrices*, Numerical Linear Algebra with Applications, Vol. 2(2), pp. 5–113 (1995).
- [17] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, 1965.
- [18] H. Zha, *A Two-Way chasing scheme for reducing a symmetric arrowhead matrix to tridiagonal form*, J. Numerical Linear Algebra, 1, pp. 494–499. 1993.

DIPARTIMENTO DI MATEMATICA E INFORMATICA, UNIVERSITÁ DEGLI STUDI DI UDINE, VIALE DELLE SCIENZE 208, I-33100 UDINE, ITALY

*E-mail address:* `fasino@dimi.uniud.it`

ISTITUTO PER LE APPLICAZIONI DEL CALCOLO "M. PICONE". CONSIGLIO NAZIONALE DELLE RICERCHE, SEZ. BARI, VIA G. AMENDOLA, 122/I, I-70126 BARI, ITALY

*E-mail address:* `N.Mastronardi@area.ba.cnr.it`

DEPARTMENT OF COMPUTER SCIENCE, KATHOLIEKE UNIVERSITEIT LEUVEN, CELESTIJNENLAAN 200A, B-3001 LEUVEN, BELGIUM

*E-mail address:* `Marc.VanBarel@cs.kuleuven.ac.be`

## A Comrade-Matrix-Based Derivation of the Eight Versions of Fast Cosine and Sine Transforms

Alexander Olshevsky, Vadim Olshevsky, and Jun Wang

*"Life as we know it would considerably different if, from 1965 Cooley-Turkey paper onwards the FFT community has made systematic and heavy use of matrix-vector notation. Indeed, couching results and algorithms in matrix-vector notation the FFT literature can be unified and made more understandable to the outsider."*

Charles Van Loan, "Computational frameworks for the FFT" [VL92]

**ABSTRACT.** The paper provides a full self-contained derivation of fast algorithms to compute discrete Cosine and Sine transforms I - IV. For the Sine I/II and Cosine I/II transforms a unified derivation based on the concept of the comrade matrix is presented. The comrade matrices associated with different versions of the transforms differ in only a few boundary elements; hence, in each case algorithms can be derived in a unified manner. The algorithm is then modified to compute Sine III/IV and Cosine III/IV transforms as well. The resulting algorithms for the versions III/IV are direct and recursive, such algorithms were missing in the existing literature. Finally, formulas reducing Cosine and Sine transforms of the types III and IV to each other are presented.

### Part I: Versions I and II Of The Transforms

#### 1. Introduction

**1.1. Preliminaries.** The FFT is, without doubt, the most important algorithm in applied mathematics and engineering. As Charles Van Loan writes in his book [VL92]: "The fast Fourier transform (FFT) is one of the truly great computational developments of this century. It has changed the face of science and engineering so that it is not an exaggeration to say that life as we know it would be very different without FFT." The importance of the FFT stems from two distinct factors: its presence in a plethora of applications, spanning almost every area of

---

1991 *Mathematics Subject Classification.* Primary 65T50, 94A12; Secondary 65F30.

*Key words and phrases.* Discrete Cosine Transform, Discrete Sine Transform, Fast Fourier Transform, Comrade Matrix, Superfast Algorithms.

This work was supported in part by NSF contracts CCR 0098222 and 0242518.

computational engineering, and the availability of fast and accurate algorithms for its computation.

The FFT uses complex arithmetic. There exist real-arithmetic analogues of the FFT, namely the Discrete Cosine Transform(DCT) and the Discrete Sine Transform(DST). Some references for these transforms can be found in [RY90] and [S99]. In this paper, the eight versions of the cosine and sine transform are considered. For these real arithmetic transforms there also exist many applications – ranging from adaptive filtering to speech and image compression. The availability of fast algorithms for their computation has led to the increasing popularity of these transforms in the last decade.

There are several different mathematical techniques which can be used to derive fast algorithms for these transforms: polynomial, matrix factorization, and others. However, none of them seem to suggest a transparent approach: algorithms for each transform are typically derived differently, with somewhat painstaking computation. In fact, there is no monograph where one could find all these transforms derived at once. Secondly, the transforms III and IV are a bit more involved than I and II (see, e.g., sec. 4 for an explanation), and for them direct recursive algorithms are missing. In this paper the latter two disadvantages are removed. In part I fast algorithms for the transforms I and II are derived via operations on companion matrices and their certain generalizations called comrade matrices. The divide-and-conquer nature of the approach causes all of the algorithms to run in  $O(n \log n)$  time. Secondly, in parts II and III new recursive algorithms are developed for the versions III and IV.

## 1.2. The Fast Fourier Transform.

1.2.1. *Definitions and the divide-and-conquer algorithm.* The Discrete Fourier Transform of a sequence  $\{u_n\}_{n=0}^{N-1}$  is the sequence  $\{U_n\}_{n=0}^{N-1}$  defined by

$$(1.1) \quad \begin{bmatrix} U_0 \\ U_1 \\ \vdots \\ U_{N-1} \end{bmatrix} = \frac{1}{\sqrt{N}} \underbrace{\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N & W_N^2 & \cdots & W_N^{N-1} \\ 1 & W_N^2 & W_N^4 & \cdots & W_N^{2(N-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & W_N^{N-1} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix}}_{F_N} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}$$

where  $W_N = e^{-\frac{2\pi i}{N}}$ . The sine and cosine transforms of the sequence  $\{u_n\}$  are defined in a similar manner, with a different matrix in place of  $F_n$ . The matrices for each transform are listed in Table 1.

	Discrete transform	Inverse transform
DCT-I	$C_N^I = \sqrt{\frac{2}{N}} \left[ \alpha_{kj} \cos \frac{kj\pi}{N-1} \right]_{k,j=0}^{N-1}$ where $\alpha_{kj} = \eta_k \eta_{N-1-k} \eta_j \eta_{N-1-j}$	$[C_N^I]^{-1} = [C_N^I]^T = C_N^I$
DCT-II	$C_N^{II} = \sqrt{\frac{2}{N}} \left[ \eta_k \cos \frac{k(2j+1)\pi}{2N} \right]_{k,j=0}^{N-1}$	$[C_N^{II}]^{-1} = [C_N^{II}]^T = C_N^{III}$
DST-I	$S_N^I = \sqrt{\frac{2}{N+1}} \left[ \sin \frac{kj}{N+1} \pi \right]_{k,j=1}^N$	$[S_N^I]^{-1} = [S_N^I]^T = S_N^I$
DST-II	$S_N^{II} = \sqrt{\frac{2}{N}} \left[ \eta_k \sin \frac{k(2j-1)}{2N} \pi \right]_{k,j=1}^N$	$[S_N^{II}]^{-1} = [S_N^{II}]^T = S_N^{III}$

Table 1. Discrete trigonometric transforms. Here,  $\eta_k = \begin{cases} \frac{1}{\sqrt{2}} & k = 0, N \\ 1 & \text{otherwise} \end{cases}$

Computation of the DFT by standard matrix-vector multiplication would take order  $O(n^2)$  operations. The Fast Fourier Transform speeds up this computation to  $O(n \log n)$  by using a divide-and-conquer approach. Namely, the FFT reduces solving the problem of size  $N$  to two problems of size  $\frac{N}{2}$  at the cost of only  $O(N)$ . Since the recursive application of this method will result in approximately  $\log N$  halving steps, the result is  $O(N \log N)$  running time. The idea behind the FFT is highlighted by the following formula (see, e.g., [S86]):

$$(1.2) \quad F_N = \begin{bmatrix} \text{odd - even} \\ \text{permutation} \end{bmatrix} \begin{bmatrix} F_{\frac{N}{2}} & 0 \\ 0 & F_{\frac{N}{2}} \end{bmatrix} \begin{bmatrix} I & I \\ \varepsilon_{\frac{N}{2}} & -\varepsilon_{\frac{N}{2}} \end{bmatrix}$$

where  $\varepsilon_{\frac{N}{2}} = \text{diag}(1, W_N, \dots, W_N^{\frac{N}{2}-1})$ . Formula (1.2) can be applied repeatedly to reduce a size- $N$  DFT's to two size- $\frac{N}{2}$  DFT, resulting in the Fast Fourier Transform.

1.2.2. *Polynomial Division Interpretation for the FFT.* Though deriving formula (1.2) for the DFT can be easily done, extending it by brute force to DCT/DST is cumbersome. Here we propose a different matrix approach which provides more insight. We first interpret the DFT as polynomial division and then express it in matrix form. In the introduction we show the result of this technique for the simplest DFT case. Then, in the main text, we spell out the details for the DCT/DST transforms.

It is possible to look at the FFT as an algorithm which computes polynomial evaluations. Indeed, if  $u(x) = x^n + u_{N-1}x^{N-1} + u_{N-2}x^{N-2} + \dots + u_1x + u_0$  then the output sequence  $\{U_k\}$  is composed of the evaluations of the polynomial  $u(x)$  at the  $N$ 'th roots of unity  $W_N^k$ :

$$(1.3) \quad \begin{bmatrix} U_0 \\ U_1 \\ \vdots \\ U_{N-1} \end{bmatrix} = F_N \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix} = \begin{bmatrix} u(1) \\ u(W_N) \\ \vdots \\ u(W_N^{N-1}) \end{bmatrix}$$

A straightforward evaluation of the polynomial  $u(x)$  takes  $O(n^2)$  operations, much like matrix-vector multiplication. However, a divide and conquer approach can be applied in order to get a faster  $O(n \log n)$  algorithm.

Let us assume that  $N$  is even and denote by  $S_0$  the  $N$ 'th roots of unity, i.e.

$$(1.4) \quad S_0 = \{1, W_N, W_N^2, \dots, W_N^{N-1}\}$$

$S_0$  can be split up into “even” and “odd” parts:

$$(1.5) \quad S_1 = \{1, W_N^2, W_N^4, \dots, W_N^{N-2}\} \quad \text{and} \quad S_2 = \{W_N, W_N^3, W_N^5, \dots, W_N^{N-1}\}$$

and associate a polynomial with each of them:

$$(1.6) \quad b_1(x) = \prod_{x_i \in S_1} (x - x_i) = x^{\frac{N}{2}} - 1 \quad \text{and} \quad b_2(x) = \prod_{x_i \in S_2} (x - x_i) = x^{\frac{N}{2}} + 1$$

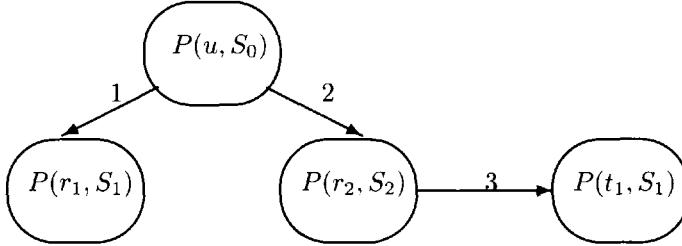
Now polynomial division can be employed

$$(1.7) \quad u(x) = b_k(x)q_k(x) + r_k(x)$$

to reduce the problem of evaluating  $u(x)$  at  $S_0$  to two problems involving the evaluation of  $r_1(x)$  at  $S_1$  and  $r_2(x)$  at  $S_2$ . Indeed, (1.6) and (1.7) imply that  $u(x_k) = r_1(x_k)$  if  $x_k \in S_1$ . Similarly,  $u(x_k) = r_2(x_k)$  if  $x_k \in S_2$ .

However, in order to accomplish  $O(N \log N)$  time, the problem needs to be reduced to the evaluation of  $\frac{n}{2} - 1$  degree polynomials at exactly the same set  $S_1$ . Hence one more step is needed: one of these lower degree evaluations must be obtained – in no more than  $O(N)$  operations – from the other.

The following picture illustrates this:



**Figure 1.** Halving the size of the problem.  $P(u, S_0)$  denotes the evaluation of  $u(x)$  at the set  $S_0$ ; other terms are defined similarly.

To derive the FFT, it remains to show how to implement branch 3 in the above illustration, i.e. how to reduce the evaluations of  $r_2(x)$  at  $S_2$  to evaluations at  $S_1$ . Since

$$\underbrace{W_N^{2k+1}}_{\in S_2} = W_N \cdot \underbrace{W_N^{2k}}_{\in S_1}$$

it follows that

$$r_2(W_N^{2k+1}) = t_1(W_N^{2k})$$

where the coefficients of  $t_1(x)$  can be obtained from the ones of  $r_2(x)$  as follows:

$$(1.8) \quad r_2(x) = a_0 + a_1x + a_2x^2 + \dots, \quad \text{and} \quad t_1(x) = a_0 + (a_1W_N)x + (a_2W_N^2)x^2 + \dots$$

We next provide a purely matrix formulation of the latter derivation. Though it might look a bit artificial at first glance, it will provide the most transparent approach to deriving algorithms for the different versions of DCT and the DST at once.

### 1.2.3. Companion Matrix Derivation of the FFT.

Let

$$(1.9) \quad A = \begin{bmatrix} 0 & I_{n-1} \\ -u_0 & -u_1, \dots, -u_{N-1} \end{bmatrix}$$

be the companion matrix of the polynomial  $u(x) = x^n + u_{N-1}x^{N-1} + u_{N-2}x^{N-2} + \dots + u_1x + u_0$ . The following well-known result shows how to divide polynomials using matrix manipulations:

**PROPOSITION 1.1.** Let  $u(x)$  and  $b(x)$  be given, and let the coefficients of  $q(x)$  and  $r(x)$  be determined by  $u(x) = q(x)b(x) + r(x)$ . Form the matrix  $[D \ E] = [I \ 0]_{(t+1, n-1)} \cdot b(A)$ , where  $t = \deg q(x)$ .

Then, the coefficients of  $q(x)$  can be obtained from:

$$(1.10) \quad [q_0 \ q_1 \ \dots \ q_{t-1} \ 1] E = 0$$

and then the coefficients of  $r(x)$  can be obtained from:

$$(1.11) \quad - [ q_0 \quad q_1 \quad \cdots \quad q_1 \quad 1 ] D = [ r_0 \quad r_1 \quad \cdots \quad r_{m-1} ]$$

where  $m = \deg b(x)$

Our goal here is to use this technique to accomplish the divisions by  $b_1(x)$  and  $b_2(x)$  in Figure 1. Since  $b_1(x)$  and  $b_2(x)$  are simple, the corresponding matrices  $X_1 = [D_1 \ E_1]$  and  $X_2 = [D_2 \ E_2]$  are simple as well:

$$(1.12) \quad X_1 = \left[ \begin{array}{cccc|ccc} -1 & 0 & \cdots & & 1 & 0 & \cdots \\ 0 & -1 & \cdots & & 0 & 1 & \cdots \\ \vdots & \cdots & \ddots & & \vdots & \ddots & \cdots \\ \vdots & & & -1 & \vdots & & 1 \\ -u_0 & -u_1 & \cdots & -u_{\frac{N}{2}-1} - 1 & -u_{\frac{N}{2}} & \cdots & -u_{N-1} \end{array} \right]$$

$$(1.13) \quad X_2 = \left[ \begin{array}{cccc|ccc} 1 & 0 & \cdots & & 1 & 0 & \cdots \\ 0 & 1 & \cdots & & 0 & 1 & \cdots \\ \vdots & \cdots & \ddots & & \vdots & \ddots & \cdots \\ \vdots & & & 1 & \vdots & & 1 \\ -u_0 & -u_1 & \cdots & -u_{\frac{N}{2}-1} + 1 & -u_{\frac{N}{2}} & \cdots & -u_{N-1} \end{array} \right]$$

Now that explicit expressions for  $X_k$  are available, it is possible to explicitly compute the remainders. Due to the sparsity of  $X_k$ , simple expressions are available. Indeed, from (1.12) it follows that  $r_1(x) = (u_{\frac{N}{2}-1} + u_{N-1})x^{\frac{N}{2}-1} + (u_{\frac{N}{2}-2} + u_{N-2})x^{\frac{N}{2}-2} + \cdots + (u_0 + u_{\frac{N}{2}})$  while (1.13) implies that  $r_2(x) = (u_{\frac{N}{2}-1} - u_{N-1})x^{\frac{N}{2}-1} + (u_{\frac{N}{2}-2} - u_{N-2})x^{\frac{N}{2}-2} + \cdots + (u_0 - u_{\frac{N}{2}})$ . From these expressions for the remainders, the decomposition (1.2) immediately follows, which is listed again for convenience:

$$(1.14) \quad F_N = \left[ \begin{array}{c} \text{odd - even} \\ \text{permutation} \end{array} \right] \left[ \begin{array}{cc} F_{\frac{N}{2}} & 0 \\ 0 & F_{\frac{N}{2}} \end{array} \right] \left[ \begin{array}{cc} I & I \\ \varepsilon_{\frac{N}{2}} & -\varepsilon_{\frac{N}{2}} \end{array} \right]$$

**1.3. Main Results.** In the rest of this paper we use the above matrix techniques to derive in a unified way four fast algorithms to compute DCT and DST of the types I and II. In each of these cases computing the fast transform means evaluation of a polynomial in the basis of Chebyshev-like polynomials. This change of basis means that instead of the companion matrix (1.9) its analogue called the comrade matrix must be used. We next highlight a modification (of the above derivation for the FFT) for the DCT-I case.

In the case of DCT-I, the transform of the sequence  $\{a_0, a_1, a_2, \dots, a_{N-1}\}$  can be reduced to the polynomial evaluations of  $a(x) = a_0 \frac{T_0}{\sqrt{2}} + a_1 T_1 + \dots + a_{N-1} \frac{T_{N-1}}{\sqrt{2}} + (xT_{N-1} - T_{N-2})$ , where  $\{T_k(x)\}$  are the Chebyshev polynomials of the first kind. For this, the comrade matrix of this polynomial can be used. This comrade matrix is as follows:

$$(1.15) \quad A = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{2} & & & & \\ & \frac{1}{2} & & \frac{1}{2} & & & \\ & & \ddots & & \ddots & & \\ & & & \frac{1}{2} & & \frac{1}{2} & \\ & & & & \frac{1}{2} & & \\ -\frac{a_0}{2} & -\frac{a_1}{2} & -\frac{a_2}{2} & \cdots & & -\frac{a_{N-2}}{2} + \frac{1}{\sqrt{2}} & -\frac{a_{N-1}}{2} \end{bmatrix}$$

The techniques described above for the FFT can now be applied to the DCT-I. The corresponding (new) set of nodes  $S_0$  can be divided into two sets, an "even" set  $S_1$  and an "odd" set  $S_2$ . The polynomial  $b_1(x)$  is chosen such that the set of roots of  $b_1(x)$  is  $S_1$ . Then,  $a(x)$  is divided by  $b_1(x)$  and the problem of evaluating  $a(x)$  at  $S_1$  is reduced to the problem of evaluating the remainder of this division at  $S_1$ . The matrices  $D_1$  and  $E_1$  (DCT-I counterparts of the ones in (1.12)) that used here to divide  $a(x)$  by  $b_1(x)$  are:

$$D_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & -\frac{\sqrt{2}}{4} & 0 \\ 0 & 0 & 0 & -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & \ddots & 0 & \frac{1}{4} & 0 \\ 0 & \frac{1}{4} & 0 & \ddots & 0 & 0 \\ -\frac{\sqrt{2}}{4} & 0 & \frac{1}{4} & 0 & 0 & 0 \\ -a_N \frac{\sqrt{2}}{4} & -a_{N-1} \frac{\sqrt{2}}{4} & \cdots & & -a_{\frac{N+1}{2}} \frac{\sqrt{2}}{4} & \end{bmatrix}$$

$$E_1 = \begin{bmatrix} \frac{\sqrt{2}}{4} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & -\frac{1}{4} & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & \ddots & 0 & 0 \\ 0 & 0 & \ddots & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & -\frac{1}{4} & 0 & \frac{\sqrt{2}}{4} \\ -a_{\frac{N-1}{2}} \frac{\sqrt{2}}{4} & \cdots & -a_3 \frac{\sqrt{2}}{4} & -a_2 \frac{\sqrt{2}}{4} & -a_1 \frac{\sqrt{2}}{4} & \end{bmatrix}$$

To compute the remainder the equations  $q \cdot E_1 = 0$  and  $r = -qD_1$  must be solved. But note that  $E_1$  is sparse! This means the equation  $q \cdot E_1 = 0$  is very easy to solve. In fact explicit formulas for the remainder – which do not require any multiplications and only  $O(N)$  additions – can be derived from these expressions for  $D_1$  and  $E_1$ .

The structure of  $D_1$  and  $E_1$  also provides a way to put the result back together once the two smaller polynomial evaluations are complete:

$$(1.16) \quad F_N = \begin{bmatrix} \text{odd} - \text{even} \\ \text{permutation} \end{bmatrix} \begin{bmatrix} F_{\frac{N+1}{2}} & 0 \\ 0 & GF_{\frac{N+1}{2}} \widehat{G} \end{bmatrix} \begin{bmatrix} I_{\frac{N+1}{2}, \frac{N+1}{2}} & \widehat{I}_{\frac{N+1}{2}, \frac{N-1}{2}} \\ \check{I}_{\frac{N-1}{2}, \frac{N+1}{2}} & -\check{I}_{\frac{N-1}{2}, \frac{N-1}{2}} \end{bmatrix}$$

where  $G$ ,  $\widehat{G}$ ,  $\widehat{I}$ ,  $\check{I}$ , and  $\widetilde{I}$  are some simple sparse matrices with  $O(N)$  complexity of multiplication. Since (1.16) reduces the problem of computing  $F_N$  to the problem of computing  $F_{\frac{N+1}{2}}$ , it can be applied repeatedly as the basis for a divide-and-conquer algorithm with the cost of  $O(N \log N)$ .

## 2. Discrete Transforms via Barnett's Comrade Matrices

The four transforms in table 1 can be reformulated in terms of the classical Chebysehev polynomials  $T_k = \cos(k \arccos x)$  and  $U_k = \frac{\sin((k+1)\arccos x)}{\sin \arccos x}$ , or their minor modifications. In this chapter, it is shown how to use this fact to derive the doubling algorithms described in the introduction.

**2.1. Polynomial Evaluation.** Let  $F_N$  be the transform matrix. i.e. the matrix  $C$  in case of the cosine transforms and  $S$  in case of the sine transforms (these  $F_N$ 's are listed in Table 1). Following [KO96] we represented them as follows:

$$(2.1) \quad F_N = W_Q \cdot V_Q$$

where

$$(2.2) \quad V_Q = \begin{bmatrix} Q_0(x_1) & Q_1(x_1) & \cdots & Q_{N-1}(x_1) \\ Q_0(x_2) & Q_1(x_2) & \cdots & Q_{N-1}(x_2) \\ \vdots & \vdots & & \vdots \\ Q_0(x_N) & Q_1(x_N) & \cdots & Q_{N-1}(x_N) \end{bmatrix}$$

and where the orthogonal( Chebyshev-like) polynomials  $Q_k(x)_{k=0}^{n-1}$  are given in Table 2, the nodes  $x_k$  are the roots of the polynomial  $Q_n$  and are given in Table 3, and finally  $W_Q$  is the weight matrix specified in Table 4 [KO96].

	$\{Q_0, Q_1, \dots, Q_{N-2}, Q_{N-1}\}$
DCT-I	$\{\frac{1}{\sqrt{2}}T_0, T_1, \dots, T_{N-2}, \frac{1}{\sqrt{2}}T_{N-1}\}$
DCT-II	$\{U_0, U_1 - U_0, \dots, U_{N-1} - U_{N-2}\}$
DST-I	$\{U_0, U_1, \dots, U_{N-1}\}$
DST-II	$\{U_0, U_1 + U_0, \dots, U_{N-1} + U_{N-2}\}$

Table 2. First  $n$  polynomials.

For each of the eight systems  $\{Q_k(x)\}_{k=0}^{N-1}$  the above Table 2 lists the first  $n$  polynomials.

To specify  $V_Q$  the nodes  $\{x_k\}_{k=1}^n$  must also be specified, or, equivalently, the last polynomial  $Q_n(x)$ , which is done in Table 3.

	$Q_N$	zeros of $Q_N$
DCT-I	$xT_{N-1} - T_{N-2}$	$\{\cos(\frac{k\pi}{N-1})\}_0^{N-1}$
DCT-II	$U_N - 2U_{N-1} + U_{N-2}$	$\{\cos(\frac{k\pi}{N})\}_0^{N-1}$
DST-I	$U_N$	$\{\cos(\frac{k\pi}{N+1})\}_1^N$
DST-II	$U_N + 2U_{N-1} + U_{N-2}$	$\{\cos(\frac{k\pi}{N})\}_1^N$

Table 3. The last polynomial  $Q_N(x)$  of DCT/DST I–IV.

DCT-I	$C_N^I = W_Q \cdot V_Q$ with $W_Q = \sqrt{\frac{2}{N-1}} \text{diag}(\frac{1}{\sqrt{2}}, 1, \dots, 1, \frac{1}{\sqrt{2}})$
DCT-II	$C_N^{II} = W_Q \cdot V_Q$ with $W_Q = \sqrt{\frac{2}{N}} \text{diag}(\frac{1}{\sqrt{2}}, \cos(\frac{\pi}{2N}), \dots, \cos(\frac{(N-1)\pi}{2N}))$
DST-I	$S_N^I = W_Q \cdot V_Q$ with $W_Q = \sqrt{\frac{2}{N+1}} \text{diag}(\sin(\frac{\pi}{N+1}), \dots, \sin(\frac{N\pi}{N+1}))$
DST-II	$S_N^{II} = W_Q \cdot V_Q$ with $W_Q = \sqrt{\frac{2}{N}} \text{diag}(\sin(\frac{\pi}{2N}), \dots, \sin(\frac{(N-1)\pi}{2N}), \frac{1}{\sqrt{2}} \sin(\frac{\pi}{2}))$

Table 4. The  $W_Q$  for DCT/DST I-IV

So the problem of computing sine and cosine transforms can be reduced to the problem of computing a matrix-vector product. Since the matrix in question is a Chebyshev-Vandermonde matrix, computing a sine or cosine transform is equivalent to evaluating a polynomial:

$$(2.3) \quad a(x) = \sum_{k=0}^{nN-1} a_k Q_k(x)$$

The only way in which this differs from the FFT is in that a different basis is used.

Now the steps described in Figure 1 should be implemented for the transforms considered in this paper. However, in order execute branches 1 and 2 in Figure 1 an appropriate way of dividing polynomials expressed in Chebyshev like bases is needed. Next, a nice algorithm for doing so is recalled.

**2.2. Barnett's Polynomial Division.** In [B84] Barnett gave a good algorithm for dividing polynomials using the comrade matrix, which will be summarized here. The advantages of this approach in are multiple: it is fast, requiring very few operations; it is convenient, as all the quantities the algorithm requires are readily available; and it is general, applying to all the transforms considered in this paper. Barnett's theorem is a direct generalization of Proposition 1.1, which was used (in the introduction) in the case of the DFT.

Before introducing Barnett's theorem, however, a few definitions must be given. Let us say a polynomial basis  $P_i(x)$  is defined by:

$$(2.4) \quad P_0(x) = 1, \quad P_1(x) = \alpha_1 x + \beta_1, \quad P_i(x) = (\alpha_i x + \beta_i)P_{i-1}(x) - \gamma_i P_{i-2}(x),$$

( $i = 2, 3, \dots, n$ ), and a polynomial  $a(x)$  monic in this basis:

$$(2.5) \quad a(x) = P_n(x) + a_1 P_{n-1}(x) + \dots + a_n P_0(x)$$

Then, the comrade matrix (a generalization of the companion matrix) of  $a(x)$  is defined [B84] to be :

$$(2.6) \quad A = \begin{bmatrix} \frac{-\beta_1}{\alpha_1} & \frac{1}{\alpha_1} & 0 & 0 & 0 \\ \frac{\gamma_2}{\alpha_2} & \frac{-\beta_2}{\alpha_2} & \frac{1}{\alpha_2} & 0 & 0 \\ 0 & \frac{\gamma_3}{\alpha_3} & \frac{-\beta_3}{\alpha_3} & \frac{1}{\alpha_3} & \dots \\ \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & \frac{\gamma_{n-1}}{\alpha_{n-1}} & \frac{-\beta_{n-1}}{\alpha_{n-1}} & \frac{1}{\alpha_{n-1}} & \frac{1}{\alpha_n} \\ \frac{-a_n}{\alpha_n} & \frac{-a_{n-1}}{\alpha_n} & \dots & \frac{-a_3}{\alpha_n} & \frac{-a_2 + \gamma_n}{\alpha_n} & \frac{-a_1 - \beta_n}{\alpha_n} \end{bmatrix}$$

It is now possible to state Barnett's theorem on polynomial division. For convenience, henceforth a polynomial monic with respect to  $P_i$  will be referred to as just monic.

**THEOREM 2.1.** [Barnett] Let  $a(x)$  be as defined above, and let  $b(x)$  be another monic polynomial of the  $m$ 'th degree. The choice  $\mu(n, m) = \frac{\alpha_{t+1}\alpha_{t+2}\cdots\alpha_n}{\alpha_1\alpha_2\cdots\alpha_m}$  makes the polynomial  $q(x)$  in

$$(2.7) \quad a(x) = \mu(n, m)[q(x)b(x) + r(x)]$$

be monic. Moreover, it is true that

$$(2.8) \quad [ q_t \quad q_{t-1} \quad \cdots \quad q_1 \quad 1 ] \cdot X + [r_{m-1}, \dots, r_1, r_0, 0, \dots, 0] = 0$$

where

$$(2.9) \quad X = [ I \quad 0 ]_{(t+1, n-1)} \cdot b(A)$$

Further, the rows of  $X$  satisfy the following recurrence relationship:

$$(2.10) \quad \rho_1 = [ b_m \quad b_{m-1} \quad \cdots \quad b_0 \quad 0 \quad \cdots \quad 0 ], \quad \rho_i = \rho_{i-1}(\alpha_{i-1}A + \beta_{i-1}I_n) - \gamma_{i-1}\rho_{i-2}$$

where

$$(2.11) \quad \rho_i = \text{the } i\text{'th row of } X, \quad \deg q(x) = t$$

A further simplification arises if it is noted that  $X$  can be split into two multiplier matrices:  $X = [ D \quad E ]$ , so that the above theorem can be adapted to compute the remainder as follows:

- (1) Solve  $q \cdot E = 0$ . Since  $E$  will be sparse, this will take very few operations.
- (2) Compute the remainder using  $-q \cdot D = r$

**2.3. Branches 1 and 2 in Figure 1.** We conclude this section with explicit formulas which can be used for the computations of remainders. In other words, numerical expressions for  $D_k$  and  $E_k$  mentioned in the previous section are provided which can be used to practically implement the algorithms described in this paper.

**2.3.1. Choice of Polynomials For DCT/DST I and II.** Given the sequence  $\{y_0, y_1, \dots, y_{N-1}\}$  it is needed to compute the polynomial evaluations of  $\sum_{k=0}^{M-1} y_k Q_k$  at the roots of  $Q_N$ . Since  $Q_N$  is obviously zero at its own roots, instead it is better to consider the slightly more convenient polynomial  $Y(x) = \sum_{k=0}^{N-1} y_k Q_k(x) + \tau Q_N$ ,

where  $\tau = \begin{cases} \frac{1}{\sqrt{2}} & \text{DCT-I or DST-I} \\ 1 & \text{DCT-II or DST-II} \end{cases}$ . This addition of a scaled  $Q_N$  makes it possible to construct the comrade matrix without scaling the original polynomial. A full list of these comrade matrices is given in following table. (It can be obtained from the fact that all families of polynomials  $\{Q_k\}$  in Tables 2 and 3 satisfy (almost, i.e., for  $k = 3, 4, \dots, n-2$  only) the same recurrence relations  $Q_k(x) = 2xQ_{k-1}(x) - Q_{k-2}(x)$  used to define the Chebyshev polynomials  $\{T_k(x)\}$ . The difference between  $\{T_k(x)\}$  and  $\{Q_k(x)\}$  is only in recurrences for the "bordering" polynomials  $Q_0(x), Q_1(x)$  and  $Q_{n-1}(x), Q_n(x)$ .

DCT-I	$\begin{bmatrix} 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\ \frac{1}{\sqrt{2}} & 0 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \ddots & 0 & 0 \\ 0 & 0 & \ddots & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{\sqrt{2}} \\ -\frac{y_0}{2} & -\frac{y_1}{2} & \dots & -\frac{y_{N-3}}{2} & -\frac{y_{N-2}}{2} + \frac{1}{\sqrt{2}} & -\frac{y_{N-1}}{2} \end{bmatrix}$
DCT-II	$\begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \ddots & 0 & 0 \\ 0 & 0 & \ddots & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{\sqrt{2}} \\ -\frac{y_0}{2} & -\frac{y_1}{2} & \dots & -\frac{y_{N-3}}{2} & -\frac{y_{N-2}}{2} + \frac{1}{2} & -\frac{y_{N-1}}{2} + \frac{1}{2} \end{bmatrix}$
DST-I	$\begin{bmatrix} 0 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \ddots & 0 & 0 \\ 0 & 0 & \ddots & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ -\frac{y_0}{2} & -\frac{y_1}{2} & \dots & -\frac{y_{N-3}}{2} & -\frac{y_{N-2}}{2} + \frac{1}{2} & -\frac{y_{N-1}}{2} \end{bmatrix}$
DST-II	$\begin{bmatrix} -\frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \ddots & 0 & 0 \\ 0 & 0 & \ddots & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{\sqrt{2}} \\ -\frac{y_0}{2} & -\frac{y_1}{2} & \dots & -\frac{y_{N-3}}{2} & -\frac{y_{N-2}}{2} + \frac{1}{2} & -\frac{y_{N-1}}{2} - \frac{1}{2} \end{bmatrix}$

Table 5. Comrade Matrices

Next, the polynomials  $b_1(x)$  and  $b_2(x)$  for the divisions in Figure 1 must be calculated. Recall that these are obtained by dividing the set of nodes into two parts, and associating a polynomial with each part. Some elementary calculation will yield the following table for  $b_1(x)$  and  $b_2(x)$ :

	$b_1(x)$	$b_2(x)$
DCT-I	$\frac{1}{2}(T_{\frac{N+1}{2}} - T_{\frac{N-1}{2}})$	$2T_{\frac{N-1}{2}}$
DCT-II	$U_{\frac{N}{2}} - 2U_{\frac{N}{2}-1} + U_{\frac{N}{2}-2}$	$U_{\frac{N}{2}} - U_{\frac{N}{2}-2}$
DST-I	$U_{\frac{N-1}{2}}$	$U_{\frac{N+1}{2}} - U_{\frac{N-3}{2}}$
DST-II	$U_{\frac{N}{2}} + 2U_{\frac{N}{2}-1} + U_{\frac{N}{2}-2}$	$U_{\frac{N}{2}} - U_{\frac{N}{2}-2}$

Table 6.  $b_1(x)$  and  $b_2(x)$  for DCT/DST I,II.

Note that in the case of DCT-I and DST-I, it is assumed that  $N$  is odd, while in the case of DCT-II and DST-II it is assumed that  $N$  is even.

**2.4. Division Matrices.** Having obtained the polynomials  $b_1(x)$  and  $b_2(x)$  associated with a step of reduction and the comrade matrices  $A$ , we now have everything needed in Theorem 2.1. Therefore, the matrices  $X_k$  associated with division using (2.11) can now be computed. Each matrix is split up into  $X = [D \ E]$ . The results are given in the tables that follow.

$D_1 =$	$\begin{bmatrix} 0 & 0 & 0 & 0 & -\frac{\sqrt{2}}{4} & 0 \\ 0 & 0 & 0 & -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & \ddots & 0 & \frac{1}{4} & 0 \\ 0 & \frac{1}{4} & 0 & \ddots & 0 & 0 \\ -\frac{\sqrt{2}}{4} & 0 & \frac{1}{4} & 0 & 0 & 0 \\ \hline -a_N \frac{\sqrt{2}}{4} & -a_{N-1} \frac{\sqrt{2}}{4} & \cdots & & -a_{\frac{N+1}{2}} \frac{\sqrt{2}}{4} & \end{bmatrix}$
$E_1 =$	$\begin{bmatrix} \frac{\sqrt{2}}{4} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & -\frac{1}{4} & 0 & 0 & 0 \\ \hline 0 & -\frac{1}{4} & 0 & \ddots & 0 & 0 \\ 0 & 0 & \ddots & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & -\frac{1}{4} & 0 & \frac{\sqrt{2}}{4} \\ \hline -a_{\frac{N-1}{2}} \frac{\sqrt{2}}{4} & \cdots & -a_3 \frac{\sqrt{2}}{4} & -a_2 \frac{\sqrt{2}}{4} & -a_1 \frac{\sqrt{2}}{4} & \end{bmatrix}$
$D_2 =$	$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & \ddots & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ \sqrt{2} & 0 & 0 & 0 & 0 & 0 \\ \hline -\sqrt{2}a_N & -\sqrt{2}a_{N-1} + 1 & -\sqrt{2}a_{N-2} & \cdots & & \end{bmatrix}$
$E_2 =$	$\begin{bmatrix} \sqrt{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & \sqrt{2} \\ \hline -a_{\frac{N+1}{2}} \sqrt{2} & -\sqrt{2}a_{\frac{N-1}{2}} & \cdots & -\sqrt{2}a_2 + 1 & -\sqrt{2}a_1 & \end{bmatrix} \quad xd$

**Table 7.** Division matrices  $X_k = [D_k \ E_k]$  for DCT-I.

$D_1 =$	$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 2 \\ 0 & 0 & 0 & -1 & 2 & -2 \\ 0 & 0 & -1 & 2 & -2 & 2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -1 & 2 & -2 & 2 & -2 & \dots \end{bmatrix}$
$E_1 =$	$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & -2 & 1 & 0 & 0 & 0 & 0 \\ -2 & 2 & -2 & 1 & 0 & 0 & 0 \\ 2 & -2 & 2 & -2 & 1 & 0 & 0 \\ -2 & 2 & -2 & 2 & -2 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & 0 \\ \dots & -a_6 + 2 & -a_5 - 2 & -a_4 + 2 & -a_3 - 2 & -a_2 + 2 & -a_1 - 1 \end{bmatrix}$
$D_2 =$	$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & \cdot & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ -a_N + 1 & -a_{N-1} & -a_{N-2} & \dots & -a_{\frac{N}{2}+1} \end{bmatrix}$
$E_2 =$	$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \cdot & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ -a_{\frac{N}{2}} & \dots & -a_2 & -a_1 + 1 \end{bmatrix}$

Table 8. Division matrices  $X_k = [D_k \ E_k]$  for DCT-II.

$D_1 =$	$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & 1 & 0 & 0 & \dots & \dots \end{bmatrix}$
$E_1 =$	$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ -a_N & -a_{N-1} + 1 & -a_{N-2} & -a_{N-3} + 1 & \dots & \dots & 0 \end{bmatrix}$
$D_2 =$	$\begin{bmatrix} 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ -a_N & -a_{N-1} & -a_{N-2} & \dots & -a_{\frac{N}{2}+1} \end{bmatrix}$
$E_2 =$	$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \cdot & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ -a_{\frac{N}{2}} & \dots & -a_2 & -a_1 \end{bmatrix}$

Table 9. Division matrices  $X_k = [D_k \ E_k]$  for DST-I.

$D_1 =$	$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 \\ 0 & 0 & 0 & 0 & 1 & 2 & 2 & 2 \\ 0 & 0 & 0 & 1 & 2 & 2 & 2 & 2 \\ 0 & 0 & 1 & 2 & 2 & 2 & 2 & 2 \\ \vdots & \vdots \\ 0 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 1 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \end{bmatrix}$
$E_1 =$	$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 2 & 1 & 0 & 0 & 0 & 0 \\ 2 & 2 & 2 & 2 & 1 & 0 & 0 & 0 \\ 2 & 2 & 2 & 2 & 2 & 1 & 0 & 0 \\ \vdots & \vdots \\ 2 & 2 & 2 & 2 & 2 & 2 & 1 & 0 \\ \dots & 0 \\ & & & & & & & 1 \end{bmatrix}$
$D_2 =$	$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -a_{N+1} & -a_{N-1} & -a_{N-2} & \dots & \dots & -a_{\frac{N}{2}+1} \end{bmatrix}$
$E_2 =$	$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ -a_{\frac{N}{2}} & \dots & -a_2 & -a_1 & -1 & \end{bmatrix}$

Table 10. Division matrices  $X_k = [D_k \ E_k]$  for DST-II.

## 2.5. Matrix Decompositions.

2.5.1. *Efficient Implementation of Division Formulas.* Having obtained the division formulas, it is now possible to derive the matrix decompositions of  $V_Q$  for all the transforms considered here. First, however, the following definitions are necessary:

Let  $V_{even}$  be the following polynomial-Chebyshev matrix:  $V_{even} = [Q_{2i}(x_{2j})]$  where

$$i, j = \begin{cases} 0, 1, 2, \dots, \frac{N}{2} - 1 & \text{in the case of DCT I /II} \\ 1, 2, \dots, \frac{N}{2} & \text{in the case of DST I/II} \end{cases}$$

Let  $V_{odd}$  is defined similarly, with the indexes now taking odd values. With these definitions, the following formula holds:

$$(2.12) \quad V_Q = \begin{bmatrix} odd - even \\ permutation \end{bmatrix} \begin{bmatrix} V_{even} & 0 \\ 0 & V_{odd} \end{bmatrix} H$$

where  $H$  can be shown to be:

	$H$
DCT-I	$\begin{bmatrix} I_{\frac{N+1}{2}, \frac{N+1}{2}} & \widehat{I}_{\frac{N+1}{2}, \frac{N-1}{2}} \\ \check{I}_{\frac{N-1}{2}, \frac{N+1}{2}} & -\check{I}_{\frac{N-1}{2}, \frac{N-1}{2}} \end{bmatrix}$
DCT-II	$\begin{bmatrix} I_{\frac{N}{2}, \frac{N}{2}} & \check{I}_{\frac{N}{2}, \frac{N}{2}} \\ I_{\frac{N}{2}, \frac{N}{2}} & -I_{\frac{N}{2}, \frac{N}{2}} \end{bmatrix}$
DST-I	$\begin{bmatrix} \check{I}_{\frac{N-1}{2}, \frac{N+1}{2}} & -I_{\frac{N-1}{2}, \frac{N-1}{2}} \\ I_{\frac{N+1}{2}, \frac{N+1}{2}} & -\widehat{I}_{\frac{N+1}{2}, \frac{N-1}{2}} \end{bmatrix}$
DST-II	$\begin{bmatrix} I_{\frac{N}{2}, \frac{N}{2}} & -\check{I}_{\frac{N}{2}, \frac{N}{2}} \\ I_{\frac{N}{2}, \frac{N}{2}} & \check{I}_{\frac{N}{2}, \frac{N}{2}} \end{bmatrix}$

Table 11. The matrix  $H$  for DCT/DST I,II.

where  $\check{I}$  is the involution matrix,  $\widehat{I}_{\frac{N+1}{2}, \frac{N-1}{2}} = \begin{bmatrix} I_{\frac{N-1}{2}, \frac{N-1}{2}} \\ 0 \end{bmatrix}$ , and  $\check{I}_{\frac{N-1}{2}, \frac{N+1}{2}} = \begin{bmatrix} I_{(\frac{N-1}{2}, \frac{N-1}{2})} & 0 \end{bmatrix}$ .

These formulas are the direct analogues of (1.2). They make it possible to decompose the problem at hand into two problems of half the size. In the next chapter, a method for obtaining  $V_{odd}$  from  $V_{even}$  will be shown – thereby producing an algorithm which achieves  $O(N \log N)$  complexity. Meanwhile, a proof of the formulas for  $H$  is provided based on the properties of the division matrices  $X_k$ .

2.5.2. *Proof.* The formulas for  $H$  are derived very similarly in all four cases. Therefore, the proof for the DCT-I case is provided here with the caveat that all of the other proofs are (almost) exactly the same.

In the case of DCT-I: let us partition  $X$  as  $X = \begin{bmatrix} \bar{X} \\ \bar{a} \end{bmatrix}$  where  $\bar{a}$  is the last row of  $X$ . This partition makes sense since, as can be seen in tables 7-10, the last row is the only one that depends on the coefficients of the polynomial being evaluated. Let us do the same thing with  $D$  and  $E$ ;  $D$  can be partitioned as  $D = \begin{bmatrix} \bar{D} \\ \bar{a}_1 \end{bmatrix}$  and  $E$  can correspondingly be partitioned as  $E = \begin{bmatrix} \bar{E} \\ \bar{a}_2 \end{bmatrix}$ .

Next, consider the equation  $[ q_t \quad q_{t-1} \quad \cdots \quad q_1 \quad 1 ] \cdot E = 0$ . Labelling  $\bar{q} = [ q_t \quad q_{t-1} \quad \cdots \quad q_1 ]$ , it can be rewritten as  $\bar{q} = -\bar{a}_2 \bar{E}^{-1}$ . Further, the equation  $q \cdot D = -r$  can be rewritten as  $\bar{q} \bar{D} + \bar{a}_1 = -r$ . Using the expression for  $\bar{q}$ , this last equation reduces to:

$$(2.13) \quad r = -(\bar{a}_1 - \bar{a}_2 \bar{E}^{-1} \bar{D})$$

However, due to the symmetry of  $X$ , the expression  $\bar{E}^{-1} \bar{D}$  reduces nicely. Indeed, using the matrices  $E_1, D_1, E_2, D_2$  from table 6, it can be seen that in the case of dividing by  $b_1(x)$ ,  $\bar{E}^{-1} \bar{D} = [\check{I} \quad 0]$  and in the case of dividing by  $b_2(x)$ ,  $\bar{E}^{-1} \bar{D} = \tilde{I}$ . Using Barnett's theorem, it is clear that  $\mu = \frac{4}{\sqrt{2}}$  in this case. From

this, two obvious expressions for the remainder coefficients follow:

$$(2.14) \quad r_1 = \begin{bmatrix} a_1 + a_N \\ a_2 + a_{N-1} \\ \vdots \\ a_{\frac{N-1}{2}} + a_{\frac{N+3}{2}} \\ a_{\frac{N+1}{2}} \end{bmatrix}, \quad r_2 = \begin{bmatrix} a_1 - a_N \\ a_2 - a_{N-1} \\ a_3 - a_{N-2} \\ \vdots \\ a_{\frac{N-1}{2}} - a_{\frac{N+3}{2}} \end{bmatrix}$$

However, a careful look at the matrix  $H$  for DCT-I in table 7 shows that it

satisfies  $H \cdot \begin{bmatrix} a_N \\ a_{N-1} \\ \vdots \\ a_1 \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \end{bmatrix}$ .

However, this is exactly the equality  $H$  needs to satisfy in order to be correct. Indeed, since the evaluations of  $a(x)$  at the "even" points are equal to the evaluations of  $r_1$ , the top half of the vector  $H\vec{a}$  after the odd-even permutation is applied should be composed of  $r_1$ . Similarly, the bottom half should be  $r_2$ . Since  $H$  satisfies this function, it follows that the formula is correct.

### 3. Branch 3

In the previous section, it has been shown how to efficiently go through branches one and two in Figure 1. In other words, it was explained how to reduce the problem of evaluating  $V_Q$  to the problem of evaluating two *different* matrices half the size:  $V_{even}$  and  $V_{odd}$ .

However, in order for the algorithms presented here to have  $O(N \log N)$  running time, it is necessary that the problem of evaluating  $V_Q$  be reduced to two exactly the same problems of half the size. This is reflected in branch three of Figure 1, where it is shown that one of these half-sized problems must be reduced to the other.

This task that is considered in this chapter. It is shown how to obtain  $V_{odd}$  from  $V_{even}$ . Namely, matrices  $G$  and  $\widehat{G}$  are found with low complexity of multiplication such that  $V_{odd} = GV_{even}\widehat{G}$ .

The approach is based on some simple trigonometric formulas which are applied to determine formulas for odd nodes in terms of even nodes.

The following table summarizes four formulas which will be used. They can be checked using direct computation.

DCT-I	$Q_k(x_{2j+1}) = \frac{Q_k(x_{2j+2}) + Q_k(x_{2j})}{2Q_k(x_1)}$
DCT-II	$Q_k(x_{2j+1}) = \frac{Q_k(x_{2j+2}) \cos(\frac{1}{2} \arccos x_{2j+2}) + Q_k(x_{2j}) \cos(\frac{1}{2} \arccos x_{2j})}{2 \cos(\frac{(2j+1)\pi}{2N}) \cos(\frac{(2k+1)\pi}{2N})}$
DST-I	$Q_k(x_{2j+1}) = \frac{Q_k(x_{2j+2}) \sin(\arccos x_{2j+2}) + Q_k(x_{2j}) \sin(\arccos x_{2j})}{2 \cos(\frac{(2j+1)\pi}{N+1}) \cos(\frac{(k+1)\pi}{N+1})}$
DST-II	$Q_k(x_{2j+1}) = \frac{Q_k(x_{2j+2}) \sin(\frac{1}{2} \arccos x_{2j+2}) + Q_k(x_{2j}) \sin(\frac{1}{2} \arccos x_{2j})}{2 \cos(\frac{(2j+1)\pi}{2N}) \cos(\frac{(2k+1)\pi}{2N})}$

Table 12. Odd nodes in terms of even nodes

Note again that  $\{Q_k\}$  is the polynomial basis for each transform and can be found in Table 2;  $x_k$  are the nodes for each transform and can be found in Table 3.

From these formulas, it is easy to derive the needed matrices  $G$  and  $\hat{G}$ . These matrices are given in Table 13. Note that in the case of DST-I, the number of odd nodes is larger than the number of even nodes. Therefore, in that case the matrices  $G$  and  $\hat{G}$  satisfy:  $G \begin{bmatrix} V_{even} & 0 \\ 0 & 1 \end{bmatrix} \hat{G} = V_{odd}$ .

DCT-I	$G = \begin{bmatrix} 1 & 1 & & & \\ & 1 & 1 & & \\ & & \ddots & \ddots & \\ & & & 1 & 1 \end{bmatrix}$ $\hat{G} = \begin{bmatrix} \frac{1}{T_0(x_1)} & & & & \\ & \frac{1}{T_1(x_1)} & & & \\ & & \ddots & & \\ & & & \frac{1}{T_{\frac{N-1}{2}-1}(x_1)} & \\ 0 & 0 & \cdots & & 0 \end{bmatrix}$
DCT-II	$G = \begin{bmatrix} \cos(\frac{a \cos x_0}{2}) & \cos(\frac{a \cos x_2}{2}) & & & \\ \cos(\frac{a \cos x_1}{2}) & \cos(\frac{a \cos x_3}{2}) & & & \\ \cos(\frac{a \cos x_2}{2}) & \cos(\frac{a \cos x_4}{2}) & & & \\ \cos(\frac{a \cos x_3}{2}) & \cos(\frac{a \cos x_5}{2}) & & & \\ & & \ddots & \ddots & \\ & & & \cos(\frac{a \cos x_{N-2}}{2}) & \\ & & & \cos(\frac{a \cos x_{N-3}}{2}) & \\ & & & \cos(\frac{a \cos x_{N-2}}{2}) & \\ & & & \cos(\frac{a \cos x_{N-1}}{2}) & \end{bmatrix}$ $\hat{G} = \frac{1}{2} \operatorname{diag}\left\{\frac{1}{\cos(\frac{a \cos x_1}{2})}, \frac{1}{\cos(\frac{a \cos x_3}{2})}, \dots, \frac{1}{\cos(\frac{a \cos x_{N-1}}{2})}\right\}$
DST-I	$G = \begin{bmatrix} \sin(a \cos x_2) & & & & \frac{(-1)^0}{\sin a \cos x_1} \\ \sin(a \cos x_1) & & & & \frac{(-1)^1}{\sin a \cos x_1} \\ \sin(a \cos x_2) & \sin(a \cos x_4) & & & \vdots \\ \sin(a \cos x_3) & \sin(a \cos x_5) & & & \\ & & \ddots & & \\ & & & \sin(a \cos x_{N-1}) & \frac{(-1)^{\frac{N-3}{2}}}{\sin a \cos x_{N-2}} \\ & & & \sin(a \cos x_{N-2}) & \frac{(-1)^{\frac{N-1}{2}}}{\sin a \cos x_N} \\ & & & \sin(a \cos x_{N-1}) & \frac{(-1)^{\frac{N-1}{2}}}{\sin a \cos x_N} \end{bmatrix}$ $\hat{G} = \frac{1}{2} \operatorname{diag}\left\{\frac{1}{T_1(x_1)}, \frac{1}{T_2(x_1)}, \dots, \frac{1}{T_{\frac{N-1}{2}}(x_1)}, 1\right\}$

Table 13. The matrices  $G$  and  $\hat{G}$

DST-II	$G = \begin{bmatrix} \sin\left(\frac{a \cos x_2}{2}\right) & & & \\ \sin\left(\frac{a \cos x_1}{2}\right) & & & \\ \sin\left(\frac{a \cos x_2}{2}\right) & \sin\left(\frac{a \cos x_4}{2}\right) & & \\ \vdots & & & \\ \sin\left(\frac{a \cos x_{N-2}}{2}\right) & & \sin\left(\frac{a \cos x_N}{2}\right) & \\ \sin\left(\frac{a \cos x_{N-1}}{2}\right) & & \sin\left(\frac{a \cos x_{N-1}}{2}\right) & \end{bmatrix}$ $\hat{G} = \frac{1}{2} \text{diag}\left\{\frac{1}{\cos\left(\frac{\pi}{2N}\right)}, \frac{1}{\cos\left(\frac{3\pi}{2N}\right)}, \frac{1}{\cos\left(\frac{5\pi}{2N}\right)}, \dots, \frac{1}{\cos\left(\frac{(N-1)\pi}{2N}\right)}\right\}$
--------	--

Table 13 Continued. The matrices  $G$  and  $\hat{G}$

**3.1. Flow Graphs.** Traditional recursive algorithms reduce the computation of  $F_N$  to the computation of  $F_{\frac{N}{2}}$ . Using the formulas obtained above, it is more natural to reduce  $V_N$  to  $V_{even}$  and use the identity  $F_N = W_N V_N$ . We provide flow graphs implementing this approach.

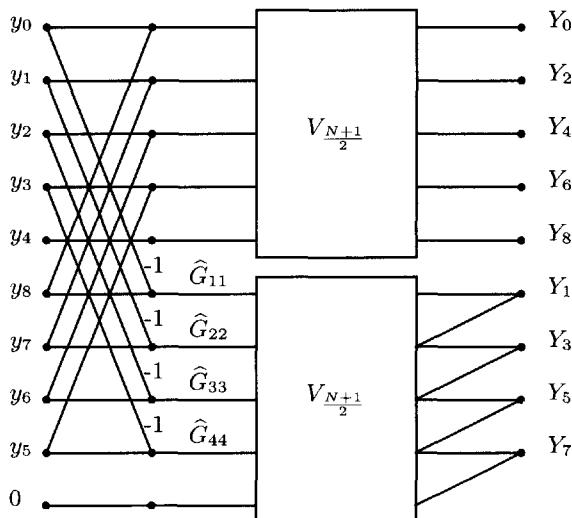
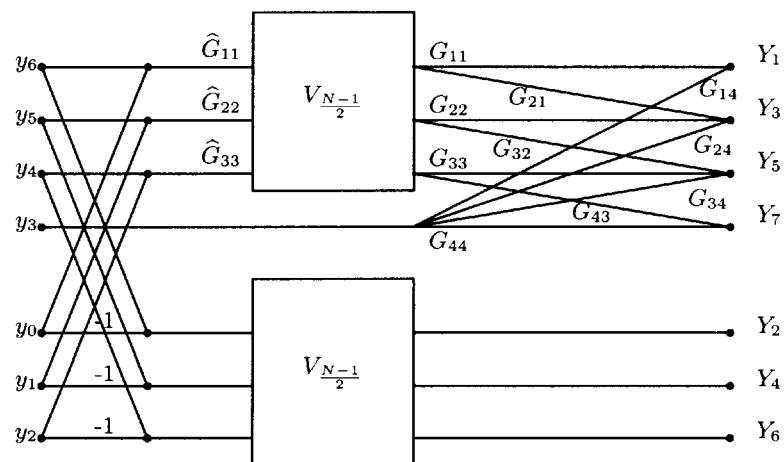
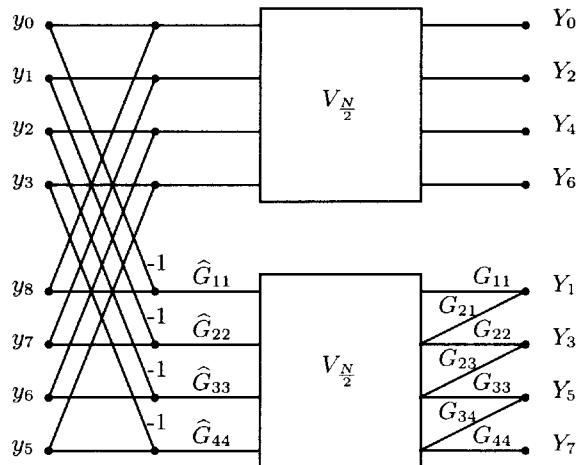


Figure 2. Flow Graph for  $V_N$  in the case of DCT-I with  $N = 9$



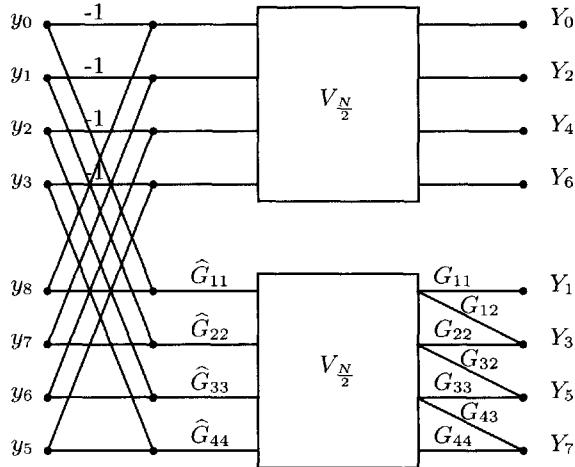


Figure 5. Flow Graph for  $V_N$  in the case of DST-II with  $N = 8$

**3.2. Passing to Recursions for the Transform.** Thus far, the following formula has been established:

$$(3.1) \quad V_Q = \begin{bmatrix} \text{odd - even} \\ \text{permutation} \end{bmatrix} \begin{bmatrix} V_{\text{even}} & 0 \\ 0 & GV_{\text{even}}\widehat{G} \end{bmatrix} H$$

This is a recursion for the Chebyshev-Vandermonde matrix  $V_Q$ . While this recursion can be used to obtain an  $O(N \log N)$  algorithm, one may wish to also see the direct recursions for  $F_N$ . As has been shown before,  $F_N = W_Q V_Q$ ; therefore, the recursions obtained can be easily modified to compute  $F_N$ . Indeed, if  $F_{\text{even}} = W_e \cdot V_{\text{even}}$  and  $F_{\text{odd}} = W_o \cdot V_{\text{odd}}$ , and if  $P = W_Q \begin{bmatrix} \text{odd - even} \\ \text{permutation} \end{bmatrix}^{-1} \begin{bmatrix} W_e^{-1} & \\ & I \end{bmatrix}$ , then it follows trivially from (3.1) that the following formula holds:

$$(3.2) \quad F_N = P \begin{bmatrix} F_{\text{even}} & 0 \\ 0 & GW_e^{-1}F_{\text{even}}\widehat{G} \end{bmatrix} H$$

where  $F_{\text{even}}$  is the transform of dimension equal to the number of even nodes in the set. In other words,

$$(3.3) \quad F_{\text{even}} = \begin{cases} F_{\frac{N+1}{2}} & \text{DCT-I} \\ F_{\frac{N-1}{2}} & \text{DST-I} \\ F_{\frac{N}{2}} & \text{DCT-II and DST-II} \end{cases}$$

## Part II: DCT III and DST III.

In this part, we will consider the versions III and IV. Their relation to Vandermonde matrices will be used to derive recursion formulas for version III transforms.

### 4. Preliminaries

At the heart of the derivation presented in Part I is Figure 1 and the formula:

$$(4.1) \quad V_Q = \begin{bmatrix} \text{odd-even} \\ \text{permutation} \end{bmatrix} \begin{bmatrix} V_{\text{even}} & 0 \\ 0 & V_{\text{odd}} \end{bmatrix} H$$

The approach described in part I could in principle be applied to the III and IV versions of the cosine and sine transforms. These transformations are defined by the transformation matrix in table 14:

DCT-III	$C_N^{III} = \sqrt{\frac{2}{N}} \left[ \eta_j \cos \frac{(2k+1)j\pi}{2N} \right]_{k,j=0}^{N-1}$	$[C_N^{III}]^{-1} = [C_N^{III}]^T = C_N^{II}$
DCT-IV	$C_N^{IV} = \sqrt{\frac{2}{N}} \left[ \cos \frac{(2k+1)(2j+1)\pi}{4N} \right]_{k,j=0}^{N-1}$	$[C_N^{IV}]^{-1} = [C_N^{IV}]^T = C_N^{IV}$
DST-III	$S_N^{III} = \sqrt{\frac{2}{N}} \left[ \eta_j \sin \frac{(2k-1)j\pi}{2N} \right]_{k,j=1}^N$	$[S_N^{III}]^{-1} = [S_N^{III}]^T = S_N^{II}$
DST-IV	$S_N^{IV} = \sqrt{\frac{2}{N}} \left[ \sin \frac{(2k-1)(2j-1)\pi}{4N} \right]_{k,j=1}^N$	$[S_N^{IV}]^{-1} = [S_N^{IV}]^T = S_N^{IV}$

Table 14. The III and IV versions of the cosine and sine transforms

These transforms are also polynomial evaluations in Chebyshev-like bases, much like their counterparts considered earlier. Tables 15,16, and 17 are borrowed from [KO96], they contain the polynomials, the nodes, and the weight matrices required to express the DCT/DST III and IV transforms in terms of Chebyshev like bases:

	$\{Q_0, Q_1, \dots, Q_{N-2}, Q_{N-1}\}$
DCT-III	$\{\frac{1}{\sqrt{2}}T_0, T_1, \dots, T_{N-1}\}$
DCT-IV	$\{U_0, U_1 - U_0, \dots, U_{N-1} - U_{N-2}\}$
DST-III	$\{U_0, U_1, \dots, U_{N-2}, \frac{1}{\sqrt{2}}U_{N-1}\}$
DST-IV	$\{U_0, U_1 + U_0, \dots, U_{N-1} + U_{N-2}\}$

Table 15. Chebyshev-like polynomials for DCT/DST III and IV

	$Q_n$	zeros of $Q_n$
DCT-III	$T_n$	$\{\cos(\frac{(2k+1)\pi}{2N})\}_{0}^{N-1}$
DCT-IV	$2T_n$	$\{\cos(\frac{(2k+1)\pi}{2N})\}_{0}^{N-1}$
DST-III	$T_n$	$\{\cos(\frac{(2k-1)\pi}{2N})\}_{1}^N$
DST-IV	$2T_n$	$\{\cos(\frac{(2k-1)\pi}{2N})\}_{1}^N$

Table 16. The last polynomial and the nodes of DCT/DST III and IV

DCT-III	$C_N^{III} = V_Q \cdot W_Q$ with $W_Q = \sqrt{\frac{2}{N}} \text{diag}(\frac{1}{\sqrt{2}}, \cos(\frac{\pi}{2N}), \dots, \cos(\frac{(N-1)\pi}{2N}))$
DCT-IV	$C_N^{IV} = W_Q \cdot V_Q$ with $W_Q = \sqrt{\frac{2}{N}} \text{diag}(\cos(\frac{\pi}{4N}), \cos(\frac{3\pi}{4N}), \dots, \cos(\frac{(2N-1)\pi}{4N}))$
DST-III	$S_N^{III} = W_Q \cdot V_Q$ with $W_Q = \sqrt{\frac{2}{N}} \text{diag}(\sin(\frac{\pi}{2N}), \dots, \sin(\frac{(N-1)\pi}{2N}), \frac{1}{\sqrt{2}} \sin(\frac{\pi}{2}))$
DST-IV	$S_N^{IV} = W_Q \cdot V_Q$ with $W_Q = \sqrt{\frac{2}{N}} \text{diag}(\sin(\frac{\pi}{4N}), \sin(\frac{3\pi}{4N}), \dots, \sin(\frac{(2N-1)\pi}{4N}))$

**Table 17.** The  $W_Q$  for DCT/DST III and IV

However, the approach presented in this paper becomes more involved when applied to the DCT/DST III and IV. One reason for this is as follows: the nodes of versions I and II are recursive whereas the nodes of versions III and IV are not. For example, in the case of DCT-II, the roots of  $Q_4$  when  $N = 4$  are a subset of the roots of  $Q_8$  when  $N = 8$ . However, table 16 shows that this is not case for versions III and IV. Hence, Figure 1 should be modified for the III and IV versions; more branches should be added and the approach presented here becomes more involved. May be this is the reason why recursive algorithms for the versions III and IV are (to the best of our knowledge) missing in the literature. To derive them a different, simpler approach can be applied as shown next.

## 5. Algorithms for Versions III of the Transforms

In order to simultaneously derive formulas for both DCT-III and DST-III, we will simply label the Chebyshev-Vandermonde matrix involved in the computation of the transform as  $V_N^{III}$ . The arguments used will apply both to the DCT and DST. Our approach rests on the following well-known fact:

$$(5.1) \quad F_N^{(III)} = F_N^{(II)}{}^T$$

where  $V_N^{II}$  is the Chebyshev-Vandermonde matrix of the corresponding version two transform(i.e. DCT-II if we are considering the transform DCT-III and DST-II if we are considering DST-III).

Because of this, we can simply take transposes of (3.1) to obtain recursions for the DCT/DST III. The end result is:

(5.2)

$$V_N^{III} = W_N^{III-1} H^T \begin{bmatrix} W_{\frac{N}{2}}^{III} V_{\frac{N}{2}}^{III} W_{\frac{N}{2}}^{III-1} & 0 \\ 0 & \widehat{G}^T W_{\frac{N}{2}}^{III} V_{\frac{N}{2}}^{III} W_{\frac{N}{2}}^{III-1} G^T \end{bmatrix} R^T W_N^{II}$$

where the quantities  $G, \widehat{G}, W_0, P$ , and  $H$  are taken from the corresponding II versions of the transforms, while  $R$  is the odd even permutation. We can rewrite this in more compact notation as:

$$(5.3) \quad V_N^{III} = L \begin{bmatrix} V_N^{\frac{III}{2}} & 0 \\ 0 & V_N^{\frac{III}{2}} \end{bmatrix} J, \quad \text{where}$$

$$L = W_N^{III^{-1}} H^T \begin{bmatrix} W_{\frac{N}{2}}^{III} & 0 \\ 0 & \tilde{G}^T W_{\frac{N}{2}}^{III} \end{bmatrix}, \quad J = \begin{bmatrix} W_{\frac{N}{2}}^{II^{-1}} & 0 \\ 0 & W_{\frac{N}{2}}^{II^{-1}} G^T \end{bmatrix} R^T W_N^{II}.$$

It is possible to obtain a recursion for the transform matrix itself. This can be done by transposing (3.2). The result is the following recursion:

$$(5.4) \quad F_N = H^T \begin{bmatrix} F_{even} & 0 \\ 0 & \tilde{G}^T F_{even} W_e^{-T} G^T \end{bmatrix} P^T$$

## 6. Flow Graphs

The following flow graphs represent the algorithms derived in the previous section. Note that each line on the right-hand side of the picture entails a multiplication which has been left off the picture due to lack of space.

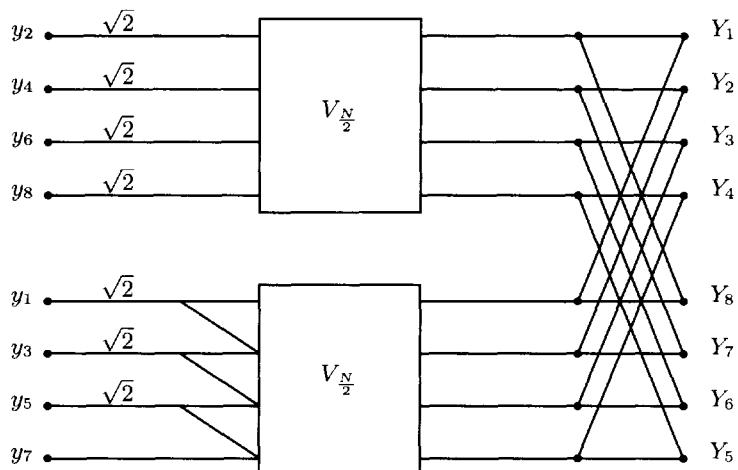


Figure 6 Flow Graph for  $V_N$  in the case of DCT-III with  $N = 8$

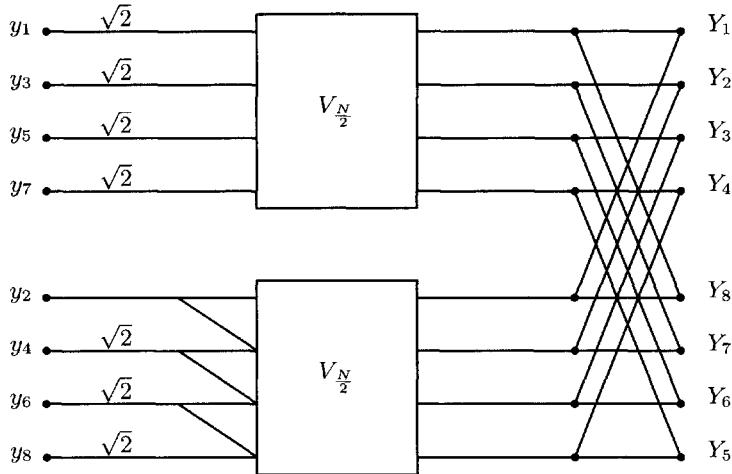


Figure 7 Flow Graph for  $V_N$  in the case of DST-III with  $N = 8$

### Part III: Reduction Formulas and Version IV

Instead of computing a particular version cosine or sine transform directly, it may be desirable to reduce it instead to the computation of a different version cosine or sine transform. In this section, formulas will be derived which do this. Furthermore, these reduction formulas will be used to easily adapt the formulas we derived earlier to the computation of version IV of the transforms.

#### 7. Reduction Formulas

One approach to the reduction of one transform to another is based on the following observation: the III and IV versions of the transforms all have the same nodes, as a quick glance at table 16 reveals. Let us label these nodes  $x_0, x_1, \dots, x_{N-1}$ .

This sharing of nodes implies that the transform matrix  $F_N$  can be expressed as follows:

$$(7.1) \quad F_N = W_N \cdot V \cdot M_N$$

where  $W_N$  and  $M_N$  depend on the transform while  $V$  does not. In all cases,  $V$  is the usual Vandermonde matrix:  $V_{ij} = [x_{i-1}^{j-1}]$ .

This factorization can easily be used to obtain the reduction formulas we seek. Indeed, let  $F_{P1}$  be the transform matrix of some transform  $P_1$ . Similarly, let  $F_{P2}$  be the transform matrix of some transform  $P_2$ . Of course,  $P_1$  and  $P_2$  can be any one of the transforms DCT/DST III and IV. We will further denote by  $W_{P1}$  and  $M_{P1}$  the matrices  $W_N$  and  $M_N$  associated with  $P_1$ ; similar notation will be used for the matrices associated with  $P_2$ .

Then,

$$(7.2) \quad F_{P1} = W_{P1} W_{P2}^{-1} F_{P2} M_{P2}^{-1} M_{P1}$$

Using the notation  $W_{P1}^{P2} = W_{P1}W_{P2}^{-1}$  and  $M_{P1}^{P2} = M_{P2}^{-1}M_{P1}$ , (7.2) can be rewritten as:

$$(7.3) \quad F_{P1} = W_{P1}^{P2}F_{P2}M_{P1}^{P2}$$

There are a total of 12 formulas compactly written in (7.3), making it possible to go from any version III or IV transform to any other version III or IV transform.

For convenience, we provide two tables of  $W_{P1}^{P2}$  and  $M_{P1}^{P2}$  for all the possibilities:

$W_{DCT\ III}^{DCT\ IV} = diag(\frac{1}{\cos(\frac{\pi}{4N})}, \frac{1}{\cos(\frac{3\pi}{4N})}, \dots, \frac{1}{\cos(\frac{(2N-1)\pi}{4N})})$
$W_{DCT\ III}^{DST\ III} = diag(\frac{1}{\sin(\frac{\pi}{2N})}, \frac{1}{\sin(\frac{3\pi}{2N})}, \dots, \frac{1}{\sin(\frac{(2N-1)\pi}{2N})})$
$W_{DCT\ III}^{DST\ IV} = diag(\frac{1}{\sin(\frac{\pi}{4N})}, \frac{1}{\sin(\frac{3\pi}{4N})}, \dots, \frac{1}{\sin(\frac{(2N-1)\pi}{4N})})$
$W_{DCT\ IV}^{DCT\ III} = diag(\cos(\frac{\pi}{4N}), \cos(\frac{3\pi}{4N}), \dots, \cos(\frac{(2N-1)\pi}{4N}))$
$W_{DCT\ IV}^{DST\ III} = diag(\frac{1}{2\sin(\frac{\pi}{4N})}, \frac{1}{2\sin(\frac{3\pi}{4N})}, \dots, \frac{1}{2\sin(\frac{(2N-1)\pi}{2N})})$
$W_{DCT\ IV}^{DST\ IV} = diag(\cotan(\frac{\pi}{4N}), \cotan(\frac{3\pi}{2N}), \dots, \cotan(\frac{(2N-1)\pi}{2N}))$
$W_{DST\ III}^{DCT\ III} = diag(\sin(\frac{\pi}{2N}), \sin(\frac{3\pi}{2N}), \dots, \sin(\frac{(2N-1)\pi}{2N}))$
$W_{DST\ III}^{DCT\ IV} = diag(2\sin(\frac{\pi}{4N}), 2\sin(\frac{3\pi}{4N}), \dots, 2\sin(\frac{(2N-1)\pi}{2N}))$
$W_{DST\ II\ I}^{DST\ IV} = diag(2\cos(\frac{\pi}{4N}), 2\cos(\frac{3\pi}{4N}), \dots, 2\cos(\frac{(2N-1)\pi}{4N}))$
$W_{DST\ IV}^{DCT\ III} = diag(\sin(\frac{\pi}{4N}), \sin(\frac{3\pi}{4N}), \dots, \sin(\frac{(2N-1)\pi}{4N}))$
$W_{DST\ IV}^{DCT\ IV} = diag(\tan(\frac{\pi}{4N}), \tan(\frac{3\pi}{4N}), \dots, \tan(\frac{(2N-1)\pi}{4N}))$
$W_{DST\ IV}^{DST\ III} = diag(\frac{1}{2\cos(\frac{\pi}{4N})}, \frac{1}{2\cos(\frac{3\pi}{4N})}, \dots, \frac{1}{2\cos(\frac{(2N-1)\pi}{4N})})$

Table 18. The Matrices  $W$  Involved in the Reduction of Transforms

$M_{DCT\ III}^{DCT\ IV} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \ddots & \ddots & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$
$M_{DCT\ III}^{DST\ III} = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & -\frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \ddots & 0 \\ 0 & 0 & 0 & \ddots & 0 & -\frac{1}{2} \\ 0 & 0 & 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} \end{bmatrix}$
$M_{DCT\ III}^{DST\ IV} = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & \ddots & \ddots & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & -\frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$
$M_{DCT\ IV}^{DCT\ III} = \begin{bmatrix} \sqrt{2} & -\sqrt{2} & \sqrt{2} & -\sqrt{2} & \sqrt{2} & \dots & -\sqrt{2} \\ 0 & 2 & -2 & 2 & -2 & \dots & 2 \\ 0 & 0 & 2 & -2 & 2 & \dots & -2 \\ 0 & 0 & 0 & 2 & -2 & \dots & 2 \\ 0 & 0 & 0 & 0 & 2 & \dots & -2 \\ 0 & 0 & 0 & 0 & 0 & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}$
$M_{DCT\ IV}^{DST\ III} = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & \ddots & \ddots & 0 \\ 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & \sqrt{2} \end{bmatrix}$

Table 19 The Matrices  $M$  Involved in the Reduction of Transforms

$M_{DCT\ IV}^{DST\ IV} =$	$\begin{bmatrix} 1 & -2 & 2 & -2 & 2 & \cdots & -2 \\ 0 & 1 & -2 & 2 & -2 & \cdots & 2 \\ 0 & 0 & 1 & -2 & 2 & \cdots & -2 \\ 0 & 0 & 0 & 1 & -2 & \cdots & 2 \\ 0 & 0 & 0 & 0 & 1 & & \vdots \\ 0 & 0 & 0 & 0 & 0 & \ddots & -2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$
$M_{DST\ III}^{DCT\ III} =$	$\begin{bmatrix} \sqrt{2} & 0 & \sqrt{2} & 0 & \sqrt{2} & \cdots & \sqrt{2} & 0 \\ 0 & 2 & 0 & 2 & \cdots & 2 & 0 & \sqrt{2} \\ 0 & 0 & 2 & 0 & \cdots & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 & 0 & \cdots & & \sqrt{2} \\ 0 & 0 & 0 & 0 & \ddots & & & \\ 0 & 0 & 0 & 0 & 0 & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sqrt{2} \end{bmatrix}$
$M_{DST\ III}^{DCT\ IV} =$	$\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 & \frac{1}{\sqrt{2}} \\ 0 & 1 & 1 & \cdots & 1 & \frac{1}{\sqrt{2}} \\ 0 & 0 & 1 & \cdots & 1 & \frac{1}{\sqrt{2}} \\ 0 & 0 & 0 & \ddots & & \vdots \\ 0 & 0 & 0 & 0 & 1 & \frac{1}{\sqrt{2}} \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} \end{bmatrix}$
$M_{DST\ III}^{DST\ IV} =$	$\begin{bmatrix} 1 & -1 & 1 & -1 & \cdots & 1 & -\frac{1}{\sqrt{2}} \\ 0 & 1 & -1 & 1 & \cdots & -1 & \frac{1}{\sqrt{2}} \\ 0 & 0 & 1 & -1 & \cdots & 1 & -\frac{1}{\sqrt{2}} \\ 0 & 0 & 0 & \ddots & & \vdots \\ 0 & 0 & 0 & 0 & 1 & -1 & \frac{1}{\sqrt{2}} \\ 0 & 0 & 0 & 0 & 0 & 1 & -\frac{1}{\sqrt{2}} \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} \end{bmatrix}$

Table 19 Continued The Matrices  $M$  Involved in the Reduction of Transforms

$M_{DST-IV}^{DCT-III} =$	$\begin{bmatrix} \sqrt{2} & \sqrt{2} & \sqrt{2} & \sqrt{2} & \sqrt{2} & \cdots & \sqrt{2} & \sqrt{2} \\ 0 & 2 & 2 & 2 & \cdots & 2 & 2 & 2 \\ 0 & 0 & 2 & 2 & \cdots & 2 & 2 & 2 \\ 0 & 0 & 0 & 2 & 2 & \cdots & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 & \ddots & & \\ 0 & 0 & 0 & 0 & 0 & 0 & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}$
$M_{DST-IV}^{DCT-IV} =$	$\begin{bmatrix} 1 & 2 & 2 & 2 & 2 & \cdots & 2 \\ 0 & 1 & 2 & 2 & 2 & \cdots & 2 \\ 0 & 0 & 1 & 2 & 2 & \cdots & 2 \\ 0 & 0 & 0 & 1 & 2 & \cdots & 2 \\ 0 & 0 & 0 & 0 & 1 & \ddots & \\ 0 & 0 & 0 & 0 & 0 & \ddots & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$
$M_{DST-IV}^{DST-III} =$	$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & \ddots & \ddots & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & \sqrt{2} \end{bmatrix}$

Table 19 Continued The Matrices  $M$  Involved in the Reduction of Transforms

### 8. Recursive Algorithms for Versions IV

In this section, recursive algorithms are derived for DCT/DST IV. The technique is based on the relationships between these transforms and the DST-III via the reduction formulas obtained earlier.

It has been shown previously that  $V^{DCT-IV} = V^{DST-III} \cdot M_{DCT-IV}^{DST-III}$ . We can use this to modify (5.3) to obtain:

$$(8.1) \quad V^{DCT-IV} = L \begin{bmatrix} V_{even}^{DCT-IV} \\ V_{even}^{DCT-IV} \end{bmatrix} \cdot \begin{bmatrix} M_{DCT-IV}^{DST-III-1} \\ M_{DCT-IV}^{DST-III-1} \end{bmatrix} \cdot J \cdot M_{DCT-IV}^{DST-III}$$

where  $L$  and  $J$  are as defined earlier. The previous equation can be modified to obtain a recursive expression for  $F_N$ :

$$(8.2) \quad F_N^{DCT-IV} = W_{DCT-IV}^{DST-III} H^T \begin{bmatrix} W_{DCT-IV, \frac{N}{2}}^{DST-III} & -1 \\ & \widehat{G}^T W_{DCT-IV, \frac{N}{2}}^{DST-III} & -1 \\ F_{\frac{N}{2}}^{DCT-IV} & & F_{\frac{N}{2}}^{DCT-IV} \end{bmatrix}.$$

$$(8.3) \quad \begin{bmatrix} M_{DCT-IV, \frac{N}{2}}^{DST-III} & -1 \\ & M_{DCT-IV, \frac{N}{2}}^{DST-III} & -1 \\ & & W_e^{-T} G^T \end{bmatrix} P^T M_{DCT-IV}^{DST-III}$$

By using the explicit expressions for the the  $W$ 's and  $M$ 's in this formula, it is possible to simplify it further. Indeed, if

$$(8.4) \quad U_{DCT-IV} = \begin{bmatrix} \sin \frac{\pi}{2N} & & & & & & \\ \sin \frac{\pi}{4N} & & & & & & \\ & \ddots & & & & & \\ & & \frac{\sin \frac{(N-1)\pi}{2N}}{\sin \frac{(N-1)\pi}{4N}} & & & & \tan \frac{\pi}{2N} \\ & & \frac{\sin \frac{(N-1)\pi}{4N}}{\sin \frac{(N-1)\pi}{2N}} & & & & \frac{\tan \frac{(N-1)\pi}{2N}}{2 \sin \frac{(N-1)\pi}{4N}} \\ & & -\frac{\sin \frac{(N+1)\pi}{2N}}{\sin \frac{(N+1)\pi}{4N}} & & & & \frac{\tan \frac{(N+1)\pi}{2N}}{2 \sin \frac{(N+1)\pi}{4N}} \\ & & & \ddots & & & \\ & & & & \frac{\tan \frac{\pi}{2N}}{2 \sin \frac{(2N-1)\pi}{4N}} & & \end{bmatrix}$$

and

$$(8.5) \quad N_{DCT-IV} = \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & \cdots & \frac{1}{\sqrt{2}} \\ & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & \cdots & \frac{1}{\sqrt{2}} \\ & & \ddots & & & & & \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & \sqrt{2} & -\sqrt{2} & \sqrt{2} & -\sqrt{2} & \cdots & \frac{1}{\sqrt{2}} \\ & & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & \sqrt{2} & -\sqrt{2} & & \\ & & & \ddots & & & & \\ & & & & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & & \end{bmatrix}$$

Then

$$(8.6) \quad F_N^{DCT-IV} = U_{DCT-IV} \begin{bmatrix} F_{\frac{N}{2}}^{DCT-IV} \\ & F_{\frac{N}{2}}^{DCT-IV} \end{bmatrix} N_{DCT-IV}$$

By the same method, a corresponding formula can be derived for DST-IV. We have that:

$$(8.7) \quad V^{DST-IV} = L \begin{bmatrix} V_{even}^{DST-IV} \\ V_{even}^{DST-IV} \end{bmatrix} \cdot \begin{bmatrix} M_{DST-IV}^{DST-III-1} \\ M_{DST-IV}^{DST-III-1} \end{bmatrix} \cdot J \cdot M_{DST-IV}^{DST-III}$$

which can be modified to get the following formula for the transform:

$$(8.8) \quad F_N^{DST-IV} = W_{DST-IV}^{DST-III} H^T \begin{bmatrix} W_{DST-IV, \frac{N}{2}}^{DST-III} & \\ & \widehat{G}^T W_{DST-IV, \frac{N}{2}}^{DST-III} \end{bmatrix} \begin{bmatrix} F_{\frac{N}{2}}^{DST-IV} \\ & F_{\frac{N}{2}}^{DST-IV} \end{bmatrix}.$$

$$(8.9) \quad \begin{bmatrix} M_{DST-IV, \frac{N}{2}}^{DST-III} & \\ & M_{DST-IV, \frac{N}{2}}^{DST-III} W_e^{-T} G^T \end{bmatrix} P^T M_{DST-IV}^{DST-III}$$

This equation can also be modified to provide recursions for the transform. If

$$(8.10) \quad U_{DST-IV} = \begin{bmatrix} \frac{\cos \frac{\pi}{2N}}{\cos \frac{\pi}{4N}} & & & & & & \\ & \ddots & & & & & \\ & & \frac{\cos \frac{(N-1)\pi}{2N}}{\cos \frac{(N-1)\pi}{4N}} & & & & \frac{1}{2 \cos \frac{\pi}{4N}} \\ & & & \frac{\sin \frac{(N-1)\pi}{2N}}{\cos \frac{(N+1)\pi}{4N}} & & & \frac{1}{2 \cos \frac{(N-1)\pi}{4N}} \\ & & & & \ddots & & \\ & -\frac{\cos \frac{\pi}{2N}}{\cos \frac{(2N-1)\pi}{4N}} & & & & \frac{1}{2 \cos \frac{(2N-1)\pi}{4N}} & \end{bmatrix}$$

and

$$(8.11) \quad N_{DCT-IV} = \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & \cdots & & \\ & & & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & \cdots & \frac{1}{\sqrt{2}} \\ & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & & & & & & \frac{1}{\sqrt{2}} \\ & & & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & & & & \\ & & & & & \ddots & & & \\ & & & & & & \frac{1}{\sqrt{2}} & & \frac{1}{\sqrt{2}} \end{bmatrix}$$

Then

$$(8.12) \quad F_N^{DST-IV} = U_{DST-IV} \begin{bmatrix} F_{\frac{N}{2}}^{DST-IV} \\ & F_{\frac{N}{2}}^{DST-IV} \end{bmatrix} N_{DST-IV}$$

These formulas can be used as a basis for recursive  $O(N \log N)$  algorithms. Indeed, there is a clear recursion for multiplication by  $U$  and  $N$ : the  $k$ 'th row can be easily computed from the  $k+1$ 'st one allowing multiplication in  $O(N)$  operations.

## 9. Flow Graphs

Flow graph implementations of the algorithms for DCT/DST IV are provided here.

Note that in the following two graphs the multiplications have been left off due to lack of space. In general each arrow entails a multiplication.

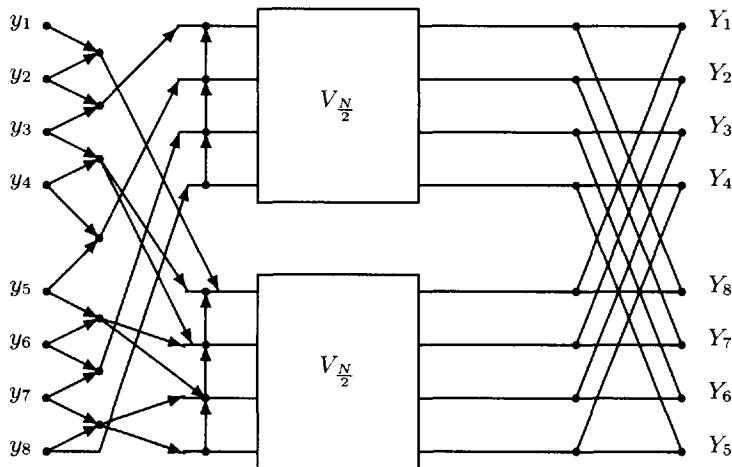


Figure 8 Flow Graph for  $V_N$  in the case of DCT-IV with  $N = 8$

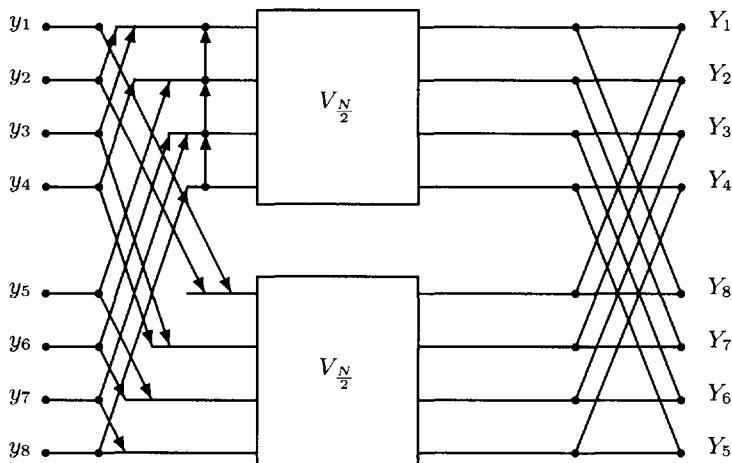


Figure 9 Flow Graph for  $V_N$  in the case of DST-IV with  $N = 8$

## References

- [B84] S. Barnett, *Division of Generalized Polynomials Using the Comrade Matrix*, Linear Algebra and Its Applications **60**:159-175 (1984)
- [KO96] T.Kailath and V.Olshevsky. *Displacement structure approach to discrete trigonometric transform based preconditioners of G.Strang and T.Chan types*. Calcolo, **33** 1996, 191-208.
- [RY90] K.R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications*, Academic Press, 1990.
- [S99] G. Strang, *The Discrete Cosine Transform*, SIAM Review, **41**:135-147.
- [S86] G. Strang, *Introduction to Applied Mathematics*, Wellesley-Cambridge Press, 1986
- [VL92] C. Van Loan, *Computational Frameworks for the Fast Fourier Transform*, SIAM Publications, 1992.

ALEXANDER OLSHEVSKY, DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING, GEORGIA INSTITUTE OF TECHNOLOGY, ATLANTA, GEORGIA 30332

*E-mail address:* alex.olshevsky@resnet.gatech.edu

VADIM OLSHEVSKY, DEPARTMENT OF MATHEMATICS, UNIVERSITY OF CONNECTICUT, STORRS, CONNECTICUT, 06269. WWW.MATH.UCONN.EDU/~ OLSHEVSK

*E-mail address:* olshevsky@math.uconn.edu

JUN WANG, DEPARTMENT OF COMPUTER SCIENCE, GEORGIA STATE UNIVERSITY, ATLANTA, GEORGIA, 30303

*E-mail address:* jjjuunn.wang@hotmail.com

*This page intentionally left blank*

## Solving certain matrix equations by means of Toeplitz computations: algorithms and applications

Dario A. Bini, Luca Gemignani and Beatrice Meini

**ABSTRACT.** We show that some classes of matrix equations can be reduced to solving a quadratic matrix equation of the kind  $X^2A + XB + C = 0$  where  $A, B, C, X$  are  $m \times m$  matrices or semi-infinite matrices. The problem of computing the minimal solution, if it exists, of the latter equation is reduced to computing the matrix coefficients  $H_0$  and  $H_{-1}$  of the Laurent matrix series  $H(z) = \sum_{i=-\infty}^{+\infty} z^i H_i$  such that  $H(z)(zA + B + z^{-1}C) = I$ . Known algorithms for this computation are revisited in terms of operations among block Toeplitz matrices and new algorithms are introduced. An application to the solution of Non-Skip-Free Markov chains is shown.

### 1. Introduction

Matrix equations are encountered in the mathematical modeling of several problems of the real world. In our analysis the main motivation comes from problems in queueing theory where the numerical solution of the Markov chains which model the problem is ultimately reduced to solving equations of the kind

$$(1.1) \quad \sum_{i=0}^n X^i A_i = 0$$

where  $n$  is a positive integer, in certain cases  $n$  may be infinite,  $A_i$ ,  $i = 0, 1, \dots$ , and  $X$  are  $m \times m$  matrices. An important feature of this kind of problems is that for a suitable positive integer  $k$ , the matrix  $I + A_k$  as well as the remaining matrices  $A_i$  have nonnegative entries and  $I + \sum_i A_i$  is a column stochastic matrix, i.e.,  $e^T \sum_i A_i = 0$ , for  $e^T = (1, 1, \dots)$ . The main interest in this framework is in the minimal nonnegative solution  $G$  of (1.1), such that  $G \geq 0$  and  $G \leq X$  for any other nonnegative solution  $X$  of (1.1). Here and hereafter, any inequality involving matrices means an entry-wise inequality. A minimal solution has also the following property: its eigenvalues are the zeros of minimum modulus of the equation  $\det(\sum_{i=0}^n z^i A_i) = 0$ . We refer the reader to the books [18], [19], [16] and to the papers [11], [12] for more details on this subject and for conditions under which  $G$  exists. In the following, “minimal” is intended in the latter sense.

---

1991 *Mathematics Subject Classification*. Primary 15A24; Secondary 65F30, 60J22.

*Key words and phrases.* Matrix equations, Toeplitz matrices, Markov chains, FFT, Graeffe iteration.

Other applications where matrix equations play an important role are damped vibrations, ladder networks, dynamic programming, stochastic filtering, statistics, and control theory. We refer the reader to [1], [23], [15], [10] for more information and references on this matter. For instance, in [10] the non linear equation

$$(1.2) \quad X + BX^{-1}A - C = 0,$$

is analyzed under the assumptions that  $B = A^T$  and both  $A$  and  $C$  are positive definite, where the main goal is to compute the extremal positive definite solutions  $X^+$ ,  $X^-$  such that  $X - X^-$  and  $X^+ - X$  are positive definite for any other positive definite solution  $X$ . A detailed analysis of this problem together with some effective solution algorithms can be found in [17], [13].

A generalization of (1.2), encountered in certain queueing problems called tree-processes [16], is the following

$$(1.3) \quad X + \sum_{i=1}^k B_i X^{-1} A_i - C = 0.$$

Algorithms for this class of equations are analyzed in [5].

In this paper we show that equations (1.2) and (1.1), with  $n \geq 2$  (included the case  $n = \infty$ ), can be reduced to solving the quadratic equation

$$(1.4) \quad \mathcal{X}^2 \mathcal{A}_2 + \mathcal{X} \mathcal{A}_1 + \mathcal{A}_0 = 0$$

where the matrices  $\mathcal{A}_2, \mathcal{A}_1, \mathcal{A}_0$  are suitably related to the blocks  $A_i$  of (1.1) and  $A, B, C$  of (1.2). Then we show that, for blocks of finite size  $m$ , the minimal solution of (1.4), that is, the one having as eigenvalues the zeros of minimum modulus of  $\det(z^2 \mathcal{A}_2 + z \mathcal{A}_1 + \mathcal{A}_0)$ , is ultimately reduced to inverting the matrix Laurent polynomial  $\mathcal{A}_0 z^{-1} + \mathcal{A}_1 + \mathcal{A}_2 z$  and we introduce some algorithms for this computation. This property is extended to the case of semi-infinite blocks. These results provide a sort of unifying framework where recent algorithms designed in the literature can be better understood [6], [7] and new ones can be easily devised. Finally we propose a new algorithm based on the evaluation/interpolation technique at the Fourier points for solving (1.4). The results are obtained by using the strict analogy between matrix Laurent polynomials and block Toeplitz matrices. We recall that a block  $n \times n$  matrix  $T = (T_{i,j})$  with  $m \times m$  blocks  $T_{i,j}$ ,  $i, j = 1, \dots, n$  is block Toeplitz if there are  $m \times m$  matrices  $S_k$ ,  $k = -n + 1, \dots, n - 1$ , such that  $T_{i,j} = S_{i-j}$ ,  $i, j = 1, \dots, n$ .

The paper is organized as follows. In section 2 we describe how the different problems can be reduced to equation (1.4). In section 3 we prove suitable relations between the minimal solution  $G$  of the equation (1.4) and the matrix coefficients  $H_i$ ,  $i \in \mathbb{Z}$  of the Laurent series  $H(z) = \sum_{i \in \mathbb{Z}} z^i H_i$  such that  $H(z)A(z) = I$ , under the condition that  $A(z) = z^{-1} \mathcal{A}_0 + \mathcal{A}_1 + z \mathcal{A}_2$  is analytic in the annulus  $\mathcal{R} = \{z \in \mathbb{C} : r_1 < |z| < r_2\}$  for  $r_1 < 1 < r_2$ . These results are used in section 4 for devising two different algorithms for computing the minimal solution of (1.4). Section 5 is devoted to an application of these tools to the numerical solution of Markov chains, where the case of Non-Skip-Free problems [11] is solved in a more computational effective way. The paper ends with section 6 devoted to the conclusions.

## 2. Reductions

In this section we show that any solution of (1.1), for any value of  $n \geq 2$  (included  $n = +\infty$ ) and of (1.2) can be obtained from a corresponding solution of (1.4) for suitable values of the block coefficients. Moreover, the minimality features are kept unchanged in the reduction from (1.1) to (1.4). We say that a solution  $G$  of (1.1) is minimal if its  $m$  eigenvalues, counted with their multiplicities, coincide with the  $m$  zeros of smallest modulus of the polynomial  $\det(\sum_{i=0}^n z^i A_i)$ . In fact, it is easy to verify that any eigenvalue of  $G$  is zero of the latter polynomial since the condition  $x^T G = \lambda x^T$  implies that  $0 = x^T (\sum_{i=0}^n G^i A_i) = x^T (\sum_{i=0}^n \lambda^i A_i)$ , i.e.,  $\det(\sum_{i=0}^n \lambda^i A_i) = 0$ . If  $n = +\infty$  the latter property still holds in the sense that if  $G$  is a solution of (1.1) then the power series  $\det(\sum_{i=0}^{\infty} z^i A_i)$  has at least  $m$  zeros which coincide with the eigenvalues of  $G$ .

The reduction of (1.2) to (1.4) is almost immediate. Multiply (1.2) to the left by  $Y = BX^{-1}$  and obtain that  $Y^2 A - YC + B = 0$ . We refer the reader to [17] for more details concerning the minimality of the solutions.

The reduction of (1.1) to (1.4) for  $2 < n < +\infty$  can be obtained in two different ways. A first technique can be derived by a trick introduced in [11], and relies on the following argument. If  $G$  is a solution of (1.1) then, by a direct inspection, we may see that

$$(2.1) \quad \mathcal{X} = \begin{bmatrix} 0 & \dots & \dots & 0 \\ \vdots & \dots & \dots & \vdots \\ 0 & \dots & \dots & 0 \\ G & G^2 & \dots & G^{n-1} \end{bmatrix}$$

is a solution of (1.4) where

$$\mathcal{A}_0 = \begin{bmatrix} 0 & \dots & \dots & 0 \\ \vdots & \dots & \dots & \vdots \\ 0 & \dots & \dots & 0 \\ A_0 & 0 & \dots & 0 \end{bmatrix}, \quad \mathcal{A}_1 = \begin{bmatrix} A_1 & A_0 & & O \\ A_2 & \ddots & \ddots & \\ \vdots & \ddots & \ddots & A_0 \\ A_{n-1} & \dots & A_2 & A_1 \end{bmatrix},$$

$$\mathcal{A}_2 = \begin{bmatrix} A_n & A_{n-1} & \dots & A_2 \\ & A_n & \ddots & \vdots \\ & & \ddots & A_{n-1} \\ O & & & A_n \end{bmatrix},$$

are matrices of size  $m(n-1)$ . Moreover, observe that the nonzero eigenvalues of  $\mathcal{X}$  are eigenvalues of  $G^{n-1}$  and any eigenvalue of  $G$  powered to  $n-1$  is eigenvalue of  $\mathcal{X}$ . Therefore if  $G$  is a minimal solution of (1.1) then  $\mathcal{X}$  in (2.1) is a minimal solution of (1.4) and vice-versa.

A different reduction can be obtained, by following [20], [8]. In fact, a direct inspection shows that if  $G$  is a solution of (1.1) then the matrix of size  $mn$

$$(2.2) \quad \mathcal{X} = \begin{bmatrix} G & G^2 & \dots & G^n \\ 0 & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & 0 \end{bmatrix}$$

solves (1.4) where

$$\mathcal{A}_0 = \begin{bmatrix} A_0 & 0 & \dots & 0 \\ 0 & 0 & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & 0 \end{bmatrix}, \quad \mathcal{A}_1 = \begin{bmatrix} A_1 & 0 & \dots & \dots & 0 \\ A_2 & I & 0 & \ddots & 0 \\ A_3 & 0 & I & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ A_n & 0 & \dots & 0 & I \end{bmatrix},$$

$$\mathcal{A}_2 = \begin{bmatrix} 0 & -I & 0 & \dots & 0 \\ 0 & -I & \ddots & & \vdots \\ \ddots & \ddots & 0 & & \\ 0 & -I & & & \\ O & 0 & & & \end{bmatrix},$$

are matrices of size  $mn$ . Moreover, if  $G$  is minimal then also  $\mathcal{X}$  of (2.2) is minimal and vice-versa since the eigenvalues of  $\mathcal{X}$  coincide with the eigenvalues of  $G$  filled up with zeros.

Unlike the former reduction, this technique can be used for reducing equation (1.1), for  $n = +\infty$ , to solving (1.4) where the matrices  $\mathcal{A}_2$ ,  $\mathcal{A}_1$  and  $\mathcal{A}_0$  become semi-infinite, that is, their entries have subscripts ranging in the set  $\mathbb{N}$  of natural numbers. For  $n = +\infty$  we assume that  $\sum_{i=0}^{+\infty} |A_i|$  has finite entries, where  $|A_i|$  is the matrix whose entries are the moduli of the entries of  $A_i$ . In fact, a direct inspection shows that if  $G$  is solution of (1.1) then the semi-infinite matrix

$$(2.3) \quad \mathcal{X} = \begin{bmatrix} G & G^2 & G^3 & \dots \\ 0 & 0 & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

is solution of  $\mathcal{A}_0 + \mathcal{X}\mathcal{A}_1 + \mathcal{X}^2\mathcal{A}_2 = 0$  where

$$(2.4) \quad \mathcal{A}_0 = \begin{bmatrix} A_0 & 0 & \dots \\ 0 & 0 & \ddots \\ \vdots & \ddots & \ddots \end{bmatrix}, \quad \mathcal{A}_1 = \begin{bmatrix} A_1 & & O \\ A_2 & I & \\ A_3 & 0 & I \\ \vdots & \vdots & \ddots & \ddots \end{bmatrix}.$$

$$\mathcal{A}_2 = \begin{bmatrix} 0 & -I & & O \\ 0 & -I & & \\ & \ddots & \ddots & \\ O & & & \ddots \end{bmatrix},$$

are semi-infinite matrices. We observe also that if the spectral radius  $\rho(G)$  of  $G$  is less than 1, then  $\mathcal{X}$  is a bounded operator in  $\ell^2(\mathbb{N})$ , i.e., in the linear space of sequences  $\mathbf{a} = \{a_i\}_{i \in \mathbb{N}}$  such that the 2-norm  $\|\mathbf{a}\| = (\sum_{i=0}^{+\infty} |a_i|^2)^{\frac{1}{2}}$  is finite. A linear operator  $L : \ell^2(\mathbb{N}) \rightarrow \ell^2(\mathbb{N})$  is bounded if its 2-norm  $\|L\| = \sup_{\|\mathbf{a}\|=1} \|L(\mathbf{a})\|$  is finite. Moreover, since  $\sum_{i=0}^{+\infty} |A_i|$  is convergent, then also  $\mathcal{A}_0$ ,  $\mathcal{A}_1$  and  $\mathcal{A}_2$  of (2.4) are bounded operators. Observe also that the nonnull eigenvalues of  $\mathcal{X}$  coincide

with the nonnull eigenvalues of  $G$ . This implies that  $G$  is minimal if and only if  $\mathcal{X}$  is a solution with eigenvalues of minimum modulus.

No reduction is known of (1.3) to (1.4). In [5] it is shown how (1.3) can be reduced to a system of quadratic equations.

### 3. Reduction to inverting a matrix Laurent polynomial

Let us consider the equation (1.1) for  $n = 2$  and for  $m < +\infty$ , define  $A(z) = (z^{-1}A_0 + A_1 + zA_2)$ , and  $a(z) = \det A(z)$ . If  $\det A_2 \neq 0$  then  $z^m a(z) = \det(A_0 + zA_1 + z^2A_2)$  is a polynomial of degree  $2m$ ; let us denote  $\xi_i$ ,  $i = 1, \dots, 2m$  its zeros and assume that

$$(3.1) \quad |\xi_1| \leq \dots \leq |\xi_m| < r_1 < 1 < r_2 < |\xi_{m+1}| \leq \dots \leq |\xi_{2m}|$$

for suitable  $r_1$ ,  $r_2$ , so that  $a(z)$  is analytic in the annulus  $\mathcal{R} = \{z \in \mathbb{C} : r_1 \leq |z| \leq r_2\}$ . If  $\det A_2 = 0$  then  $z^m a(z)$  is a polynomial with degree  $k < 2m$  and we keep denoting  $\xi_1, \dots, \xi_{2m}$  its zeros where we assume that  $\xi_{k+1}, \dots, \xi_{2m}$  are zeros at infinity. If  $G$  is a solution such that  $\rho(G) = |\xi_m|$  then  $\xi_1, \dots, \xi_m$  are the eigenvalues of  $G$  and we have the block Wiener-Hopf factorization

$$(3.2) \quad \begin{aligned} A(z) &= (I - z^{-1}G)(B_0 - zB_1) = U(z^{-1})L(z) \\ B_1 &= -A_2, \quad B_0 = GA_2 + A_1, \end{aligned}$$

where  $\det U(z^{-1})$  is analytic for  $|z| > r_1$ , and  $\det L(z)$  is analytic for  $|z| < r_2$ . Equation (3.2) can be rewritten in matrix form as

$$(3.3) \quad \begin{bmatrix} A_1 & A_0 & O \\ A_2 & A_1 & A_0 \\ O & \ddots & \ddots & \ddots \end{bmatrix} = \begin{bmatrix} I & -G & O \\ & I & -G \\ O & & \ddots & \ddots \end{bmatrix} \begin{bmatrix} B_0 & O \\ -B_1 & B_0 \\ O & \ddots & \ddots \end{bmatrix} = UL$$

where  $U$  and  $L$  are block upper triangular and lower triangular semi-infinite block Toeplitz matrices, respectively.

**PROPOSITION 3.1.** *Assume that the polynomial  $\det(zA(z))$  has zeros  $\xi_i$ ,  $i = 1, \dots, 2m$  such that (3.1) holds so that  $a(z) = \det A(z)$  is analytic in the annulus  $\mathcal{R} = \{z \in \mathbb{C} : r_1 \leq |z| \leq r_2\}$ . Assume that there exists a matrix  $G$  such that  $\rho(G) < 1$  and*

$$G^2 A_2 + GA_1 + A_0 = 0,$$

*so that the eigenvalues of  $G$  coincide with the  $m$  zeros of  $z^m a(z)$  of modulus less than  $r_1$ . Then the following properties hold*

- (1)  $\det B_0 \neq 0$ , for  $B_0 = GA_2 + A_1$ .
- (2) There exist, bounded, the operators  $U^{-1}$ ,  $L^{-1}$  of (3.3)
- (3) There exist the Laurent matrix polynomial  $H(z) = \sum_{i=-\infty}^{+\infty} z^i H_i$  such that  $A(z)H(z) = I$
- (4) For  $S = B_0^{-1}B_1 = -(GA_2 + A_1)^{-1}A_2$ , it holds  $\rho(S) < 1/r_2 < 1$ ,

$$A_2 + A_1 S + A_0 S^2 = 0,$$

the sum  $\sum_{i=0}^{+\infty} G^i B_0^{-1} S^i$  is finite and

$$H_i = \begin{cases} H_0 G^{-i}, & i < 0, \\ \sum_{i=0}^{+\infty} G^i B_0^{-1} S^i & i = 0, \\ S^i H_0 & i > 0. \end{cases}$$

(5) *If the matrix equation*

$$Y^2 A_0 + YA_1 + A_2 = 0$$

*has a solution  $R$  such that  $\rho(R) < 1/r_2 < 1$ , then  $\det H_0 \neq 0$ ,  $\det(YA_0 + A_1) \neq 0$  and  $W = -(YA_0 + A_1)^{-1}A_0$  solves the equation*

$$A_2 W^2 + A_1 W + A_0 = 0.$$

*moreover we have the following representation of  $H_i$*

$$H_i = \begin{cases} H_0 G^{-i} &= W^{-i} H_0, & i \leq 0, \\ H_0 R^i &= S^i H_0, & i \geq 0 \end{cases}$$

*therefore,  $G = H_0^{-1}H_{-1}$ ,  $R = H_0^{-1}H_1$ ,  $S = H_1 H_0^{-1}$ ,  $W = H_{-1} H_0^{-1}$ .*

PROOF. From (3.2) we deduce that  $\det A(z) = \det(I - z^{-1}G) \det(B_0 - zB_1)$ . Therefore, since  $G$  has eigenvalues of modulus less than 1 which coincide with  $\xi_1, \dots, \xi_m$ , then  $L(z) = B_0 - zB_1$  is singular only for  $z = \xi_i$ ,  $i = m+1, \dots, 2m$ , whence we deduce that  $B_0 = L(0)$  is nonsingular and  $\rho(S) < 1/r_2 < 1$  for  $S = B_0^{-1}B_1$ . Moreover, the upper block triangular semi-infinite block Toeplitz matrix  $Z$  defined by its first block row  $(I, G, G^2, \dots)$  is such that  $ZU = UZ = I$  and, since  $\rho(G) < r_1 < 1$ ,  $Z$  is a bounded operator and  $Z = U^{-1}$ . A similar argument applies to the lower block triangular block Toeplitz semi-infinite matrix  $V$  defined by its first column  $(B_0^{-1}, SB_0^{-1}, S^2B_0^{-1}, \dots)$ , for  $S = B_0^{-1}B_1$ , since it holds that  $VL = LV = I$  and  $\rho(S) < 1/r_2 < 1$ . Part (3) follows with

$$H(z) = (I - zS)^{-1}B_0^{-1}(I - z^{-1}G)^{-1}$$

which is analytic in the annulus  $\mathcal{R}$ . From the latter equation we deduce that

$$(3.4) \quad H_k = \begin{cases} S^k (\sum_{i=0}^{+\infty} S^i B_0^{-1} G^i) = S^k H_0 & \text{for } k > 0, \\ \sum_{i=0}^{+\infty} S^i B_0^{-1} G^i & \text{for } k = 0, \\ (\sum_{i=0}^{+\infty} S^i B_0^{-1} G^i) G^{-k} = H_0 G^{-k} & \text{for } k < 0. \end{cases}$$

The sum  $H_0 = \sum_{i=0}^{+\infty} S^i B_0^{-1} G^i$  has entries with finite values since, for any operator norm  $\|\cdot\|$  it holds  $\|H_0\| \leq \sum_{i=0}^{+\infty} \|S^i\| \cdot \|B_0^{-1}\| \cdot \|G^i\|$ : moreover, from the relations between spectral radius and operator norms we know that for any  $\epsilon > 0$  there exists a norm  $\|\cdot\|$  such that  $\|G\| \leq \rho(G) + \epsilon$  and from the continuity of norms, since  $\lim_i S^i = 0$ , we deduce that  $\|S^i\|$  is bounded from above by a constant  $\gamma$ . Whence we obtain that  $\|H_0\| \leq \gamma \|B_0^{-1}\| \sum_{i=0}^{+\infty} (\rho(G) + \epsilon)^i$  which is bounded from above by a constant if  $\epsilon$  is such that  $\rho(G) + \epsilon < 1$ . Concerning part (5), if there exists a solution  $R$  of the equation  $R^2 A_0 + RA_1 + A_2 = 0$  such that  $\rho(R) < 1$  then, by following the same arguments as before, we deduce that  $YA_0 + A_1$  is nonsingular and the matrix  $W = -(YA_0 + A_1)^{-1}A_0$  solves the equation  $A_2 W^2 + A_1 W + A_0 = 0$ . In particular we obtain that  $H_i = H_0 R^i$  for  $i \geq 0$  and  $H_i = W^{-i} H_0$  for  $i < 0$ , so that  $H(z) = H_0 (I + \sum_{i=1}^{+\infty} (z^{-i} G^i + z^i R^i)) = (I + \sum_{i=1}^{+\infty} (z^{-i} W^i + z^i S^i)) H_0$ . The singularity of  $H_0$  would imply that  $\det H(z) = 0$  for any  $z \in \mathcal{R}$  which is absurd since for  $z \in \mathcal{R}$  it holds  $\det H(z) = 1/\det A(z) \neq 0$ .  $\square$

By using the equivalence of norms over a finite dimensional space and using the relation between operator norms and spectral radius, we may deduce that for

any operator norm  $\|\cdot\|$  there exists a constant  $\gamma > 0$  such that

$$(3.5) \quad \|H_i\| \leq \begin{cases} \|H_0\| \cdot \|G^{-i}\| \leq \gamma r_1^{-i} & i \leq 0 \\ \|H_0\| \cdot \|R^i\| \leq \gamma r_2^{-i} & i \geq 0 \end{cases}$$

that is, the coefficients of  $H_i$  decay exponentially to zero. This nice property allows us to look at  $H(z)$  as to a Laurent polynomial plus some truncation error

$$(3.6) \quad H(z) = \sum_{i=-N/2}^{N/2-1} z^i H_i + E(z),$$

where the norm of the block coefficients of  $E(z)$  is  $O(\rho^{N/2})$  for  $\rho = \max\{r_1, r_2^{-1}\}$ . If  $N$  is “large enough” the coefficients  $H_{-N/2}, \dots, H_{N/2-1}$  can be approximated by means of the coefficients of the Laurent matrix polynomial

$$\tilde{H}(z) = \sum_{i=-N/2}^{N/2-1} z^i \tilde{H}_i$$

which interpolates  $H(z)$  at the  $N = 2^d$  roots of 1, i.e., such that

$$(3.7) \quad \begin{cases} \tilde{H}(\omega_N^j) = H(\omega_N^j), & j = 0, 1, \dots, N-1 \\ H(\omega_N^j) = [A(\omega_N^j)]^{-1} \end{cases}$$

where  $\omega_N = \cos(2\pi/N) + i \sin(2\pi/N)$  and  $i^2 = -1$ .

In fact, we have the following estimate for the infinity norm  $\|H_i - \tilde{H}_i\|_\infty$ .

**PROPOSITION 3.2.** *For the coefficients  $\tilde{H}_i$  of the Laurent polynomial  $\tilde{H}(z)$  satisfying (3.7) it holds  $\|H_i - \tilde{H}_i\|_\infty \leq \max_j \|E(\omega_N^j)\|_\infty$ . Therefore  $\|H_i - \tilde{H}_i\|_\infty \leq \alpha \rho^{N/2}$  for a positive constant  $\alpha$  and for  $\rho = \max\{r_1, r_2^{-1}\}$ .*

**PROOF.** From (3.6) it holds  $\sum_{j=-N/2}^{N/2-1} (\tilde{H}_j - H_j) \omega_N^{ij} = E(\omega_N^i)$ ,  $i = -N/2, \dots, N/2-1$ , whence  $(\tilde{H}_j - H_j) = \frac{1}{N} \sum_{i=-N/2}^{N/2-1} \omega_N^{-ij} E(\omega_N^i)$ ,  $j = -N/2, \dots, N/2-1$ . Taking the infinity norm of both sides of the latter equation yields  $\|\tilde{H}_j - H_j\|_\infty \leq \max_{i=-N/2, \dots, N/2-1} \|E(\omega_N^i)\|_\infty$ . Since  $E(z) = \sum_{i \geq N/2} z^i H_i + \sum_{i \leq -N/2-1} z^i H_i$ , from (3.5) we conclude that there exists a constant  $\alpha$  such that  $\|E(\omega_N^i)\|_\infty \leq \alpha \rho^{N/2}$  for  $\rho = \max\{r_1, r_2^{-1}\}$ .  $\square$

**3.1. The case of semi-infinite blocks.** The case where the blocks  $A_0, A_1, A_2$  are semi-infinite is more tricky as the following example shows. Let  $Z = (z_{i,j})$ ,  $z_{i+1,i} = 1$ ,  $z_{i,j} = 0$  for  $i \neq j+1$  be a semi-infinite matrix and let  $A_0 = -2I$ ,  $A_1 = Z$ ,  $A_2 = Z^2$ . Since  $Z^T Z = I$ , then  $X = Z^T$  solves the equation  $A_0 + X A_1 + X^2 A_2 = 0$ . On the other hand the spectrum of  $X$  is  $\mathcal{S}_X = \{\lambda \in \mathbb{C} : |\lambda| \leq 1\}$ , whereas the set of values of  $\lambda$  such that  $A_0 + \lambda A_1 + \lambda^2 A_2$  is not invertible is  $\mathcal{S} = \{\lambda \in \mathbb{C} : |\lambda| \geq 1\}$ . Therefore  $\lambda \in \mathcal{S}_X$  does not necessarily imply that  $\lambda \in \mathcal{S}$ .

However, under suitable assumptions, we may arrive at a characterization of the minimal solution of the quadratic equation  $A_0 + X A_1 + X^2 A_2 = 0$  in terms of the blocks  $H_0$  and  $H_{-1}$  of the Laurent series  $H(z) = \sum_{i \in \mathbb{Z}} z^i H_i$  such that  $H(z)A(z) = A(z)H(z) = I$ . In fact now we prove that (3.4) still holds in the case of semi-infinite blocks.

Let  $\mathcal{R} = \{z \in \mathbb{C} : r_1 \leq |z| \leq r_2\}$  for given  $r_1 < 1$  and  $r_2 > 1$ , denote  $\rho = \max\{r_1, r_2^{-1}\}$ . Assume that  $A_0, A_1, A_2$  are semi-infinite bounded operators in  $\ell^2(\mathbb{N})$  such that the operator Laurent polynomial  $A(z) = z^{-1}A_0 + A_1 + zA_2$  is analytic and invertible with bounded inverse for  $z \in \mathcal{R}$ . In this hypothesis there exists an operator Laurent series  $H(z) = \sum_{i \in \mathbb{Z}} z^i H_i$  which is analytic and bounded for  $z \in \mathcal{R}$  such that  $H(z)A(z) = A(z)H(z) = I$ ; moreover it holds

$$H_k = \frac{1}{2\pi i} \int_{|z|=r} z^{-1-k} A(z)^{-1} dz$$

where  $r_1 \leq r \leq r_2$  (see [21]). So that, for the 2-norm  $\|\cdot\|$ , there exists a positive constant  $\alpha$  such that

$$(3.8) \quad \|H_k\| \leq \alpha \rho^{|k|}, \text{ for } k \in \mathbb{Z}.$$

Assume also that the operator represented by the semi-infinite matrix

$$(3.9) \quad \mathcal{K} = \begin{bmatrix} A_1 & A_0 & O \\ A_2 & A_1 & A_0 \\ O & \ddots & \ddots & \ddots \end{bmatrix}$$

is invertible and there exists a semi-infinite matrix  $G$  which solves the equation  $G^2 A_2 + G A_1 + A_0 = 0$  and such that  $\|G^k\| \leq \beta g^k$  for a positive constant  $\beta$  and a positive  $g < 1$ . Then the following factorization holds

$$(3.10) \quad \mathcal{K} = \begin{bmatrix} I & -G & O \\ & I & -G \\ O & \ddots & \ddots \end{bmatrix} \begin{bmatrix} B_0 & O \\ -B_1 & B_0 \\ O & \ddots & \ddots \end{bmatrix},$$

where  $B_0 = GA_2 + A_1$ ,  $B_1 = -A_2$ . We have the following

**LEMMA 3.3.** *The inverse of  $\mathcal{K}$  can be represented as*

$$\mathcal{K}^{-1} = \begin{bmatrix} H_0 & H_{-1} & H_{-2} & \dots \\ H_1 & H_0 & H_{-1} & \ddots \\ \vdots & \ddots & \ddots & \ddots \end{bmatrix} - \begin{bmatrix} H_1 \\ H_2 \\ \vdots \end{bmatrix} \begin{bmatrix} G & G^2 & \dots \end{bmatrix}.$$

**PROOF.** Multiply the above expression to the right by  $\mathcal{K}$ , use the fact that  $[G, G^2, \dots] \mathcal{K} = [-A_0, 0, \dots]$  and that  $H_{i+1}A_0 + H_iA_1 + H_{i-1}A_2$  is zero for  $i \neq 0$  and is the identity matrix for  $i = 0$ , and obtain that the result of this product is the identity matrix.  $\square$

From the factorization (3.10) we find that  $B_0$  is invertible and, setting  $S = B_0^{-1}B_1$  we have

$$\mathcal{K}^{-1} \begin{bmatrix} I & -G & O \\ & I & -G \\ O & \ddots & \ddots \end{bmatrix} \begin{bmatrix} B_0 & O \\ B_0 & \ddots \\ O & \ddots \end{bmatrix} = \begin{bmatrix} I & & O \\ S & I & \\ S^2 & S & I \\ \vdots & \ddots & \ddots & \ddots \end{bmatrix}$$

that is, in the light of Lemma 3.3, we get  $S^k = H_k B_0 - H_{k+1} G B_0$ . Whence, from (3.8) it follows that  $\|S^k\| \leq \delta \rho^k$  for a positive  $\delta$ . This implies the invertibility of  $I - zS$  for  $z \in \mathcal{R}$  and from the relation  $H(z) = (I - zS)^{-1} B_0^{-1} (I - z^{-1}G)^{-1}$  we deduce that (3.4) holds.

Now, if we assume also that  $R$  is a bounded solution of the equation  $R^2A_0 + RA_1 + A_2 = 0$  such that  $\|R^k\| \leq \theta r^k$  for positive  $\theta$  and  $r < 1$ , and that the semi-infinite matrix

$$(3.11) \quad \mathcal{H} = \begin{bmatrix} A_1 & A_2 & O \\ A_0 & A_1 & A_2 \\ O & \ddots & \ddots & \ddots \end{bmatrix}$$

is invertible, then, by following the same argument used above to prove the validity of (3.4), we arrive at the expression  $H_i = H_0 R^i$ , for  $i \geq 0$ . Whence we obtain that

$$H(z) = H_0(I + \sum_{i=1}^{+\infty} (z^{-i}G^i + z^iR^i)).$$

The invertibility of  $H_0$  follows from the latter equation and from the invertibility of  $\Phi(z) = I + \sum_{i=1}^{+\infty} (z^{-i}G^i + z^iR^i)$  for  $z \in \mathcal{R}$ . In order to prove the invertibility of  $\Phi(z)$  we observe that multiplying  $\Phi(z)$  on the left by  $I - Rz$  we get

$$(I - Rz)\Phi(z) = (I - RG)\sum_{i=0}^{+\infty} G^i z^{-i} = (I - RG)(I - Gz^{-1})^{-1}.$$

Therefore,  $\Phi(z) = (I - Rz)^{-1}(I - RG)(I - Gz^{-1})^{-1}$  is invertible as product of invertible operators.

The invertibility of  $H_0$  allows us to obtain the following representations of the semi-infinite matrices  $G$  and  $R$

**THEOREM 3.4.** *Let  $A_0$ ,  $A_1$  and  $A_2$  be semi-infinite bounded operators on  $\ell^2(\mathbb{N})$  such that the operator Laurent polynomial  $A(z) = z^{-1}A_0 + A_1 + zA_2$  is analytic and invertible with bounded inverse for  $z \in \mathcal{R}$ . Let  $H(z) = \sum_{i \in \mathbb{Z}} z^i H_i = A(z)^{-1}$ . Moreover, assume that  $\mathcal{K}$  and  $\mathcal{H}$  in (3.9), (3.11) are invertible and there exist semi-infinite matrices  $G$  and  $R$  such that  $G^2A_2 + GA_1 + A_0 = 0$ ,  $R^2A_0 + RA_1 + A_2 = 0$ , and  $\|G^k\| \leq \beta g^k$ ,  $\|R^k\| \leq \theta r^k$ , for positive constants  $\beta$ ,  $\theta$  and positive  $g, r < 1$ . Then we have*

$$\begin{aligned} G &= H_0^{-1}H_{-1} \\ R &= H_0^{-1}H_1. \end{aligned}$$

Similar arguments can be applied for the solutions  $S$  and  $W$  of the matrix equations  $A_2 + A_1S + A_0S^2 = 0$  and  $A_2W^2 + A_1W + A_0 = 0$ .

#### 4. Algorithms

From Propositions 3.1 and 3.2 of section 3 we obtain the following algorithm for computing a minimal solution of the equation (1.1) for  $n = 2$ .

**ALGORITHM 4.1.** Solution of a quadratic matrix equation.

**INPUT:** The  $m \times m$  matrices  $A_0, A_1, A_2$  such that the Laurent polynomial  $A(z) = z^{-1}A_0 + A_1 + zA_2$  satisfies the assumptions of Proposition 3.1.

**OUTPUT:** An approximation  $\tilde{G}$  of a minimal solution  $G$  of the matrix equation  $A_0 + XA_1 + X^2A_2 = 0$ .

**COMPUTATION:**

**STAGE 1.** Let  $N$  a sufficiently large integer and denote  $\omega_N$  a primitive  $N$ -th root of 1. Compute  $W_i = A(\omega_N^i)$ ,  $i = -N/2, \dots, N/2 - 1$ .

**STAGE 2.** Compute  $V_i = W_i^{-1}$  for  $i = -N/2, \dots, N/2 - 1$ .

STAGE 3. Compute  $\tilde{H}_i = \frac{1}{N} \sum_{j=-N/2}^{N/2-1} \omega_N^{-ij} V_j$ ,  $i = -1, 0$ .

STAGE 4. Output the approximation  $\tilde{G} = \tilde{H}_0^{-1} \tilde{H}_{-1}$ .

It is immediate to verify that the cost of the above algorithm is  $\gamma_1 m^3 N + \gamma_2 m^2 N$  for two suitable constants  $\gamma_1$  and  $\gamma_2$ . Moreover, due to Proposition 3.2 it is sufficient to chose  $N = O(\log \epsilon^{-1} / \log \rho^{-1})$  for  $\rho = \max\{r_1, 1/r_2\}$  in order to have an approximation error at most  $\epsilon$  in the solution.

Algorithm 4.1 can be implemented with a heuristic strategy for the automatic evaluation of the value of the integer  $N$ . In fact, it is possible to apply the above algorithms with different values of  $N$ , say,  $N = N_0, 2N_0, 4N_0, \dots$ , until the difference of two consecutive approximations of the solution has norm less than a given error bound  $\epsilon$ . The quantities computed by the algorithm applied with  $N = 2^k N_0$  can be used in the computation with  $N = 2^{k+1} N_0$ .

**REMARK 4.2** (On the case of structured blocks). Observe that if the quadratic equation is obtained through the reduction of the general equation (1.1) to (1.4), then the blocks  $A_i$  are structured. Therefore, the cost of stage 1 of the above algorithm is substantially reduced. In fact, these blocks belong to a linear space whose dimension is much lower than the square of the size of the blocks and the linear combinations involved in stage 1 can be simply restricted to a lower dimensional basis. We can get computational advantages even in stage 2 and 3 if the structure is somehow kept under inversion. A detailed analysis of this case is performed in section 5.1 concerning an application to the numerical solution of certain Markov chains.

**REMARK 4.3** (On the case of semi-infinite blocks). In principle the above algorithm can be applied even in the case of semi-infinite blocks and the matrices have the structure (2.4). In fact, in this case in order to compute  $G$  we have to compute only the leading principal submatrix of size  $m \times m$  of  $\mathcal{X}$  of (2.3). For this purpose we have to compute all the entries of  $H_0$  and the first  $m$  columns of  $H_{-1}$ . A crucial issue is the inversion of the semi-infinite matrices  $W_i = A_0 \omega_N^{-i} + A_1 + A_2 \omega_N^i$ ,  $i = 0, 1, \dots, N-1$ , which is needed at stage 2 as well as the computation of  $H_0^{-1}$ . Indeed, these computations must be done in approximate way and the analysis of the conditions on  $A(z)$  in order that the inverse of a finite section of  $W_i$  (or of  $H_0$ ) provides the sought approximation are under investigation.

**4.1. Using Graeffe's iteration.** Assume that  $m$  is finite and consider the following sequence of Laurent matrix polynomials  $\{A^{(i)}(z)\}_{i \in \mathbb{N}}$

$$(4.1) \quad \begin{aligned} A^{(0)}(z) &= A(z) = z^{-1} A_0 + A_1 + z A_2 \\ A^{(i+1)}(z^2) &= A^{(i)}(z) [A_1^{(i)}]^{-1} A^{(i)}(-z), \quad i = 0, 1, \dots, \end{aligned}$$

where  $A^{(i)}(z) = z^{-1} A_0^{(i)} + A_1^{(i)} + z A_2^{(i)}$ . Under the assumptions of Proposition 3.1, if the matrices  $A_1^{(i)}$ ,  $i = 0, 1, \dots$ , are nonsingular then the following properties hold

(compare [4]):

$$G^{2^i} \text{ solves } A_0^{(i)} + XA_1^{(i)} + X^2A_2^{(i)} = 0,$$

$$R^{2^i} \text{ solves } X^2A_0^{(i)} + XA_1^{(i)} + A_2^{(i)} = 0,$$

$$\|A_0^{(i)}\| = O(r_1^{2^i}), \rho(G) < r_1 < 1,$$

$$\|A_2^{(i)}\| = O(r_2^{-2^i}), \rho(R) < r_2^{-1} < 1,$$

$$\|A_1^{(i)}\|, \|A_1^{(i)}\|^{-1} \leq \gamma, \gamma > 0,$$

$$\lim_i A^{(i)}(z)[A_1^{(i)}]^{-1} = I.$$

Moreover we have the following identity which can be proved by induction on  $i$

$$A^{(i+1)}(z^{2^{i+1}})[A_1^{(i+1)}]^{-1} = A(z)H^{(i)}(z)$$

$$H^{(i)}(z) = \left\{ [A_1]^{-1}A(-z)[A_1^{(1)}]^{-1}A^{(1)}(-z^2)[A_1^{(2)}]^{-1}A^{(2)}(-z^4) \dots \right. \\ \left. A^{(i)}(-z^{2^i}) \right\} [A_1^{(i+1)}]^{-1}$$

whence we find that  $H^{(i)}(z)$  converges to  $H(z)$  double exponentially.

The following algorithm approximates the minimal solution  $G$  of the equation  $G^2A_2 + GA_1 + A_0 = 0$  by means of the relation  $G = H_0^{-1}H_{-1}$  where  $H_0$  and  $H_{-1}$  are approximated by means of Graeffe's iteration.

#### ALGORITHM 4.4. Solution of a quadratic matrix equation.

**INPUT:** The  $m \times m$  matrices  $A_0, A_1, A_2$  such that the assumptions of Proposition 3.1 are satisfied; a positive real  $\epsilon$ .

**OUTPUT:** An approximation of the minimal solution  $G$  of the equation  $G^2A_2 + GA_1 + A_0 = 0$ .

#### COMPUTATION:

**STAGE 1.** (Graeffe iteration) Compute the Graeffe sequence  $A^{(i)}(z)$   $i = 0, \dots, k+1$  of (4.1) until  $\|A_0^{(k+1)}\| \leq \epsilon, \|A_2^{(k+1)}\| \leq \epsilon$ .

#### STAGE 2. (Back substitution)

$$B^{(k)}(z) = A^{(k)}(-z)[A_1^{(k+1)}]^{-1}$$

$$B^{(i-1)}(z) = A^{(i-1)}(-z)[A_1^{(i)}]^{-1}B^{(i)}(z^2), i = k, \dots, 1.$$

#### STAGE 3. Output $[B_0^{(0)}]^{-1}B_{-1}^{(0)}$ .

Observe that for computing the central blocks  $B_{-1}^{(i-1)}, B_0^{(i-1)}, B_1^{(i-1)}$ , of  $B^{(i-1)}(z)$  with the above algorithm, only the central blocks  $B_{-1}^{(i)}, B_0^{(i)}, B_1^{(i)}$  of  $B^{(i)}(z)$  are needed. It can be shown that the cost per step is just one matrix inversion, 10 matrix multiplications and 4 matrix additions.

This algorithm is strictly related with the one introduced in [6] which is based on the cyclic reduction technique. Both the algorithms compute the coefficients of the Graeffe sequence. Algorithm 4.4 performs the back substitution stage separately whereas the algorithm of [6] performs the back substitution stage simultaneously with Graeffe's iteration.

**REMARK 4.5** (On the case of semi-infinite blocks). In principle, the above algorithm can be applied even in the case  $m = +\infty$ . A crucial issue is the approximation of the inverses of the semi-infinite matrices  $A_1^{(i)}$  in the Graeffe sequence, with  $A_1^{(0)} = A_1$ , as well as the inversion of  $B_0^{(0)}$ . The analysis of conditions of  $A(z)$

under which this approximation can be performed by means of finite sections is under investigation.

## 5. The case of Markov chains

For matrix equations arising from the solution of Markov chains there is an additional difficulty. Namely, the polynomial  $\det(zA(z))$  is zero for  $z = 1$  so that the assumptions of Proposition 3.1 are not satisfied anymore. More precisely, the zeros of  $\det(zA(z))$  are

$$|\xi_1| \leq \cdots \leq \xi_m = 1 < |\xi_{m+1}| \leq \cdots \leq |\xi_{2m}|$$

so that  $\rho(G) = 1$ , moreover,  $e^T G = e^T$ ,  $e^T = (1, \dots, 1)$  (cf. [18]). (Without loss of generality we may also assume that 1 is the only zero of  $\det(zA(z))$  of modulus 1). This drawback can be easily overcome by applying a simple idea introduced in [14] for accelerating the convergence rate of iterative algorithms for computing  $G$ .

Let  $u \in \mathbb{R}^m$ ,  $e^T u = 1$  and set  $V = G - ue^T$ . The eigenvalues of  $V$  are  $\xi_1, \dots, \xi_{m-1}, 0$ , in fact,  $e^T V = e^T G - e^T = 0$ , and if  $w \neq 0$  is a right eigenvector of  $G$ , i.e., such that  $Gw = \lambda w$ ,  $\lambda \neq 1$ , then  $w$  is orthogonal to the left eigenvector  $e$ , i.e.,  $w^T e = 0$ . Therefore  $Vw = Gw - ue^T w = Gw = \lambda w$ , that is,  $\lambda$  is eigenvalue of  $V$ . Moreover it holds  $G^i = V^i + \sum_{j=0}^{i-1} V^j ue^T$ . Thus the equation  $A_0 + GA_1 + G^2 A_2 = 0$  is reduced to

$$(5.1) \quad \hat{A}_0 + V\hat{A}_1 + V^2\hat{A}_2 = 0,$$

where

$$\begin{aligned} \hat{A}_0 &= A_0 + ue^T(A_1 + A_2), \\ \hat{A}_1 &= A_1 + ue^T A_2, \\ \hat{A}_2 &= A_2. \end{aligned}$$

Furthermore  $\det(\hat{A}_0 + z\hat{A}_1 + z^2\hat{A}_2)$  has exactly  $m$  zeros of modulus  $< 1$  so that the hypotheses of Proposition 3.1 are satisfied by the equation (5.1) and we may apply the machinery of section 4 in order to approximate  $V$  and then recover  $G$  as  $G = V + ue^T$ .

**5.1. Non-Skip-Free Markov chains.** A special case encountered in queueing problems is the class of Non-Skip-Free Markov chains [11] where we are given integers  $n, m, k$  and  $m \times m$  matrices  $A_0, \dots, A_n$  such that  $A_k + I \geq 0$ ,  $A_i \geq 0$ ,  $i \neq k$ , and  $\sum_{i=0}^n A_i + I$  is stochastic, that is  $e^T \sum_{i=0}^n A_i = 0$ . For this class of problems we define  $A(z) = z^{-k} \sum_{i=0}^n z^i A_i$ .

For simplicity, we assume that  $\det A_0 \neq 0$  and  $\det A_n \neq 0$ . These assumptions can be relaxed by means of a continuity argument. Also in this case, if  $\det A_n = 0$  we extend the number of zeros of  $\det(z^k A(z))$  to  $mn$  by adding zeros at infinity. Another assumption that does not reduce the generality of this analysis is that 1 is the only zero of  $\det(z^k A(z))$  of modulus 1 [11]. Under these assumptions we find that the polynomial  $\det(z^k A(z))$  has

- $mk - 1$  zeros of modulus  $< 1$
- one zero equal to 1
- $(n - k)m$  zeros of modulus  $> 1$ .

Moreover, there exists the Wiener-Hopf factorization

$$(5.2) \quad A(z) = \left( \sum_{i=0}^k z^{-i} U_i \right) \left( \sum_{i=0}^{n-k} z^i L_i \right) = U(z^{-1}) L(z)$$

where  $U_i$  and  $L_i$  are  $m \times m$  matrices such that  $U_0 = I$ ,  $\det(\sum_{i=0}^k z^i U_i)$  has  $mk$  zeros of modulus  $\geq 1$ ,  $\det(\sum_{i=0}^{n-k} z^i L_i)$  has  $m(n-k)$  zeros of modulus  $> 1$ .

Here the problem is to compute the matrices  $U_0, \dots, U_k$  in (5.2).

Now we show that we may reduce our study to the case where the polynomial  $\det(z^k A(z))$  is nonzero for  $z = 1$ . In fact, similarly to the case of a quadratic equation, we may apply a deflation technique for removing the unit zero of  $\det(z^k A(z))$ . We show the generalization of this technique below.

Let  $\mathbf{u}$  be any vector such that  $\mathbf{u}^T \mathbf{e} = 1$  and consider the matrix  $E = \mathbf{u} \mathbf{e}^T$  such that  $E^i = E$ ,  $i \neq 0$ . Define

$$E(z) = \sum_{i=0}^{+\infty} z^i E^i = I + E \sum_{i=1}^{+\infty} z^i$$

and observe that  $E(z) = (I - zE)^{-1}$ . Since  $\mathbf{e}^T \sum_{i=0}^n A_i = 0$ , it holds that  $E \sum_{i=0}^n A_i = 0$ . Therefore the function  $\widehat{A}(z) = E(z^{-1}) A(z)$  is a matrix Laurent polynomial, i.e.,

$$(5.3) \quad \begin{aligned} \widehat{A}(z) &= z^{-k} \sum_{i=0}^n \widehat{A}_i z^i, \\ \widehat{A}_i &= A_i + E \left( \sum_{j=i+1}^n A_j \right), \quad i = 0, \dots, n. \end{aligned}$$

We may also write

$$A(z) = (I - z^{-1} E) \widehat{A}(z).$$

Now, for the sake of simplicity assume that  $A_0$  and  $A_n$  are nonsingular (the case where  $\det A_0 = 0$  or  $\det A_n = 0$  can be treated by means of a continuity argument). Since  $\widehat{A}_n = A_n$  the polynomials  $\det(z^k \widehat{A}(z))$  and  $\det(z^k A(z))$  have the same degree, and, therefore the same number of roots.

Since the matrix  $I - z^{-1} E$  is defined for  $z \neq 0$  and is singular only for  $z = 1$ , we deduce that if  $\lambda$  is zero of  $\det(z^k \widehat{A}(z))$  and  $\lambda \neq 0, 1$ , then it is zero of  $\det(z^k A(z))$  and vice-versa. Moreover,  $\det \widehat{A}(0) = 0$  since  $\mathbf{e}^T \widehat{A}(0) = \mathbf{e}^T \widehat{A}_0 = \mathbf{e}^T (\sum_{i=0}^n A_0) = 0$ . Thus, we obtain that  $\det(z^k \widehat{A}(z))$  has the same zeros of  $\det(z^k A(z))$  except for  $z = 1$  which is replaced in  $\det(z^k \widehat{A}(z))$  by  $z = 0$ .

Now observe that multiplying equation  $A(z) = U(z^{-1}) L(z)$  to the left by  $E(z)^{-1}$  yields

$$\begin{aligned} \widehat{A}(z) &= \widehat{U}(z^{-1}) L(z) \\ \widehat{U}(z) &= E(z) U(z) = \sum_{i=0}^k z^i \widehat{U}_i \\ \widehat{U}_i &= U_i + E \sum_{j=0}^{i-1} U_j, \quad i = 0, \dots, k. \end{aligned}$$

Therefore, in order to compute  $U_0, \dots, U_k$  of the factorization (5.2), we may first evaluate the blocks  $\widehat{A}_i$ ,  $i = 0, \dots, n$  by means of (5.3), then compute the Wiener-Hopf factorization  $\widehat{A}(z) = \widehat{U}(z^{-1})L(z)$  of  $\widehat{A}(z)$  and recover  $U_i$ ,  $i = 0, \dots, k$ , from the blocks  $\widehat{U}_i$ ,  $i = 0, \dots, k$ , by means of the equation  $U(z) = E(z)^{-1}\widehat{U}(z)$ , i.e..

$$\begin{aligned} U_0 &= \widehat{U}_0 \\ U_i &= \widehat{U}_i - E\widehat{U}_{i-1}, \quad i = 1, \dots, k. \end{aligned}$$

Observe that the Laurent polynomial  $\det \widehat{A}(z)$  is nonzero for  $z = 1$ . Therefore it is no loss of generality to assume that the zeros  $\xi_i$  of  $\det(zA(z))$  are such that

$$(5.4) \quad |\xi_1| \leq \dots \leq |\xi_{mk}| < 1 < |\xi_{mk+1}| \leq \dots \leq |\xi_m|.$$

Now let us rewrite  $A(z) = U(z^{-1})L(z)$  in matrix form as

$$(5.5) \quad \left[ \begin{array}{ccccc} A_k & A_{k-1} & \dots & A_0 & O \\ A_{k+1} & A_k & A_{k-1} & \ddots & A_0 \\ \vdots & \ddots & \ddots & \ddots & \ddots \\ A_n & \ddots & \ddots & \ddots & \ddots \\ O & \ddots & \ddots & \ddots & \ddots \end{array} \right] = \left[ \begin{array}{ccccc} U_0 & U_1 & \dots & U_k & O \\ U_0 & U_1 & \ddots & U_k & \\ O & \ddots & \ddots & \ddots & \ddots \end{array} \right] \left[ \begin{array}{ccccc} L_0 & & & & O \\ L_1 & L_0 & & & \\ \vdots & L_1 & L_0 & & \\ L_{n-k} & \ddots & \ddots & \ddots & \\ O & \ddots & \ddots & \ddots & \ddots \end{array} \right]$$

If we partition the matrices in (5.5) into  $qm \times qm$  blocks, where  $q = \max\{k, n - k\}$ , we obtain the equivalent equation

$$\left[ \begin{array}{ccc} \mathcal{A}_1 & \mathcal{A}_0 & O \\ \mathcal{A}_2 & \mathcal{A}_1 & \mathcal{A}_0 \\ \ddots & \ddots & \ddots \\ O & \ddots & \ddots \end{array} \right] = \left[ \begin{array}{cccc} I & -\mathcal{G} & & O \\ & I & -\mathcal{G} & \\ O & & \ddots & \ddots \end{array} \right] \left[ \begin{array}{ccc} \mathcal{B}_0 & & O \\ -\mathcal{B}_1 & \mathcal{B}_0 & \\ O & \ddots & \ddots \end{array} \right]$$

where

$$\begin{aligned} \mathcal{A}_0 &= \left[ \begin{array}{cc} A_{k-q} & O \\ \vdots & \ddots \\ A_{k-1} & \dots & A_{k-q} \end{array} \right], \quad \mathcal{A}_1 = \left[ \begin{array}{ccc} A_k & \dots & A_{k-q+1} \\ \vdots & \ddots & \vdots \\ A_{k+q-1} & \dots & A_k \end{array} \right], \\ \mathcal{A}_2 &= \left[ \begin{array}{cc} A_{k+q} & \dots & A_{k+1} \\ \ddots & & \vdots \\ O & & A_{k+q} \end{array} \right] \end{aligned}$$

and

$$\mathcal{G} = - \begin{bmatrix} U_q & & O \\ \vdots & \ddots & \\ U_1 & \dots & U_q \end{bmatrix} \begin{bmatrix} U_0 & \dots & U_{q-1} \\ & \ddots & \vdots \\ O & & U_0 \end{bmatrix}^{-1},$$

$$\mathcal{B}_0 = \begin{bmatrix} U_0 & \dots & U_{q-1} \\ & \ddots & \vdots \\ O & & U_0 \end{bmatrix} \begin{bmatrix} L_0 & O \\ \vdots & \ddots \\ L_{q-1} & \dots & L_0 \end{bmatrix},$$

$$\mathcal{B}_1 = - \begin{bmatrix} U_0 & \dots & U_{q-1} \\ & \ddots & \vdots \\ O & & U_0 \end{bmatrix} \begin{bmatrix} L_q & \dots & L_1 \\ & \ddots & \vdots \\ O & & L_q \end{bmatrix},$$

where we assume that  $A_i = 0$ ,  $U_i = 0$ ,  $L_i = 0$  if  $i$  is out of range.

A simple way for solving this computational problem is to rewrite it in terms of matrix equation as

$$\mathcal{A}_0 + \mathcal{G}\mathcal{A}_1 + \mathcal{G}^2\mathcal{A}_2 = 0$$

and to apply the algorithms of section 4 for its solution under the assumption (5.4).

It is important to point out that it is enough to compute the first block column of the matrix  $\mathcal{G}$  which provides the sought solution  $U_1, \dots, U_k$ .

For the algorithm 4.1 based on the evaluation/interpolation technique, we have the following simplifications.

- Due to the structure of  $\mathcal{A}_0$ ,  $\mathcal{A}_1$ ,  $\mathcal{A}_2$ , the evaluation of  $W_i = \omega_N^{-i}\mathcal{A}_0 + A_1 + \omega_N^i\mathcal{A}_2$  for  $i = 0, \dots, N-1$  at stage 1 can be restricted to the first  $m$  columns and to the first  $m$  rows so that the cost is  $O(Nm^2q)$  ops.
- Since the matrices  $W_i$ ,  $i = 0, \dots, N-1$  are  $\omega_N^{-i}$ -block circulant matrices [7], then their inversion at stage 2 can be performed by means of FFTs and costs  $O(Nm^2q \log q + Nm^3)$  ops. We recall that the set of  $\alpha$ -block circulant matrices is the algebra formed by matrices of the kind  $\sum_{i=0}^{q-1} C(\alpha)^i \otimes P_i$ , where

$$C(\alpha) = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & \ddots & \ddots & 1 \\ \alpha & 0 & \dots & 0 & 0 \end{bmatrix},$$

$\otimes$  denotes the Kronecker product and  $P_i$  are  $m \times m$  matrices.

- Since  $\omega_N^{-i}$ -block circulant matrices are an algebra made up by block Toeplitz matrices, then the matrices  $V_i = W_i^{-1}$  are block Toeplitz and also the matrices  $\tilde{H}_i$  at stage 4 are block Toeplitz. Hence, their computation costs  $O(Nm^2q)$  ops.
- Finally, at stage 3 we have to compute only the first  $m$  columns of  $\tilde{X}$  and this computation can be accomplished by solving  $m$  block Toeplitz systems. The cost of this part is independent of  $N$ .

The overall cost of Algorithm 4.1 amounts to  $O(N(qm^3 + m^2q \log q))$ , where we ignore the part independent of  $N$ , which for  $k = O(n)$  reduces to  $O(nm^3 + nm^2 \log n)$ . This approach compares favorably with the customary techniques based

on functional iterations [11] where the computational cost is  $O(m^3n^2)$  ops per step. The acceleration that we obtain this way is by a factor of  $n$ .

Similar accelerations can be obtained if the Graeffe iteration is applied by using the “Toeplitzness” of the blocks  $H_{-1}, H_0, H_1$ .

Another possibility of dealing with Non-Skip-Free Markov chains is to apply the reduction based on (2.4). The structure analysis of the matrices  $\mathcal{A}_0$ ,  $\mathcal{A}_1$ , and  $\mathcal{A}_2$  obtained in this way has been done only for finite blocks [8].

The last remark of this section concern the case where  $\det A(z)$  has more than one zero of modulus 1. If  $\xi \neq 1$  is such that  $|\xi| = 1$  and  $\det A(\xi) = 0$ , and if we know a vector  $\mathbf{w}$  such that  $\mathbf{w}^T A(\xi) = 0$ , then we find that  $\widehat{\mathbf{w}}^T \widehat{A}(\xi) = 0$ , for  $\widehat{\mathbf{w}}^T = \mathbf{w}^T(I - \xi^{-1}E)$ , therefore we may apply the same deflation technique in order to deflate the root  $\xi$  from the matrix Laurent polynomial  $\widehat{A}(z)$ . That is, it is enough to chose a vector  $\mathbf{v}$  such that  $\mathbf{v}^T \widehat{\mathbf{w}} = 1$ , define the matrix  $E_1 = \mathbf{v} \widehat{\mathbf{w}}^T$  and the function  $E_1(z) = (I - zE_1)^{-1}$  and finally consider the deflated matrix Laurent polynomial  $E_1(z^{-1})\widehat{A}(z)$ .

## 6. Conclusions

We have seen that different classes of matrix equations can be reduced to solving a quadratic matrix polynomial equation. The computation of the minimal solution of a quadratic matrix polynomial equation is reduced to inverting a matrix Laurent polynomial. The same approach applies to quadratic matrix equations with semi-infinite blocks. Two algorithms for the computation of the minimal solution have been introduced. One is based on the evaluation/interpolation technique at the roots of 1, the second one is based on applying the Graeffe iteration to matrix Laurent polynomials. An application to solving problems related to the numerical solution of Markov chains described by a matrix Laurent polynomial  $A(z)$  has been shown. A technique for deflating the unwanted unit zero of  $\det A(z)$  has been presented and extended for the computation of the Wiener-Hopf factorization in the Non-Skip-Free model.

**Acknowledgment.** The authors wish to thank two anonymous referees who have given a great contribution to improve the presentation of the paper.

## References

- [1] W. N. Anderson Jr., T. D. Morley, and G. E. Trapp. Positive solutions to  $X = A - BX^{-1}B^*$ . *Linear Algebra Appl.*, 134:53–62, 1990.
- [2] S. Barnett. *Polynomials and Linear Control Systems*. Number 77 in Textbooks in Pure and Applied Mathematics. Marcel Dekker, Inc., New York, 1983.
- [3] D. A. Bini and A. Böttcher. Polynomial factorization through Toeplitz matrix computations. *Linear Algebra Appl.*, to appear.
- [4] D. A. Bini, L. Gemignani, and B. Meini. Computations with infinite Toeplitz matrices and polynomials. *Linear Algebra Appl.*, 343/344 (2002), 21–61.
- [5] D. A. Bini, G. Latouche, and B. Meini. Solving nonlinear matrix equations arising in Tree-Like stochastic processes, *Linear Algebra Appl.*, to appear.
- [6] D. A. Bini and B. Meini. On the solution of a nonlinear matrix equation arising in queueing problems. *SIAM J. Matrix Anal. Appl.*, 17:906–926, 1996.
- [7] D. A. Bini and B. Meini. Solving block banded block Toeplitz systems with structured blocks: algorithms and applications, in *Structured Matrices: recent Developments in Theory and*

*Computation*, D.A. Bini, E. Tyrtyshnikov and P. Yalamov, Editors, Nova Science Publisher Inc., New York 2001.

- [8] D. A. Bini, B. Meini, and V. Ramaswami. Analyzing M/G/1 paradigms through QBDs: the role of the block structure in computing the matrix  $G$ . In G. Latouche and P. Taylor, editors, *Advances in Algorithmic Methods for Stochastic Models*, pages 73–86. Notable Publications, New Jersey, USA, 2000. Proceedings of the Third Conference on Matrix Analytic Methods.
- [9] A. Böttcher and B. Silbermann. *Introduction to Large Truncated Toeplitz Matrices* (Universitext, Springer-Verlag, New York, 1999).
- [10] A. Ferrante and B. C. Levy. Hermitian solutions of the equation  $X = Q + NX^{-1}N^*$ . *Linear Algebra Appl.*, 247:359–373, 1996.
- [11] H. R. Gail, S. L. Hantler, and B. A. Taylor. Non-skip-free M/G/1 and G/M/1 type Markov chains. *Adv. Appl. Prob.*, 29:733–758, 1997.
- [12] H. R. Gail, S. L. Hantler, and B. A. Taylor. Use of characteristics roots for solving infinite state Markov chains. In W. K. Grassmann, editor, *Computational Probability*, pages 205–255. Kluwer Academic Publishers, 2000.
- [13] C.-H. Guo and P. Lancaster. Iterative solution of two matrix equations. *Math. Comp.* 68 (1999), 1589–1603.
- [14] C. He, B. Meini, and N. H. Rhee. A shifted cyclic reduction for quasi birth-death problems, *SIAM J. Matrix Anal. Appl.*, 23 (2001/02), 673–691.
- [15] N. J. Higham and H.-M. Kim. Numerical analysis of a quadratic matrix equation. *IMA J. of Numerical Analysis*, 20:499–519, 2000.
- [16] G. Latouche and V. Ramaswami. *Introduction to Matrix Analytic Methods in Stochastic Modeling*. ASA-SIAM Series on Statistics and Applied Probability 5. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1999.
- [17] B. Meini. Efficient computation of the extreme solutions of  $X + A^*X^{-1}A = Q$ ,  $X - A^*X^{-1}A = Q$ . *Math. Comp.*, 71 (2002), 1189–1204.
- [18] M. F. Neuts. *Structured Stochastic Matrices of M/G/1 Type and Their Applications*. Dekker Inc., 1989.
- [19] M. F. Neuts. *Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach*. The Johns Hopkins University Press, Baltimore, MD, 1981.
- [20] V. Ramaswami. The generality of Quasi Birth-and-Death processes, in *Advances in Matrix Analytic Methods for Stochastic Models*, A. Alfa and S. Chakravarthy Editors, pp. 93–113, Notable Publications Inc., NJ 1998.
- [21] A. E. Taylor and D. C. Lay. *Introduction to Functional Analysis*, John Wiley& Sons, New York 1980.
- [22] X. Zhan and J. Xie. On the matrix equation  $X + A^TX^{-1}A = I$ . *Linear Algebra Appl.*, 247:337–345, 1996.
- [23] X. Zhan. Computing the extremal positive definite solutions of a matrix equation. *SIAM J. Sci. Comput.*, 17:1167–1174, 1996.

DIPARTIMENTO DI MATEMATICA, UNIVERSITÀ DI PISA, ITALY.

E-mail address: bini@dm.unipi.it, gemignan@dm.unipi.it, meini@dm.unipi.it

*This page intentionally left blank*

# A Fast Singular Value Algorithm for Hankel Matrices

Franklin T. Luk and Sanzheng Qiao

**ABSTRACT.** We present an  $O(n^2 \log n)$  algorithm for finding all the singular values of an  $n$ -by- $n$  complex Hankel matrix. We take advantage of complex symmetry and the Hankel structure. Our method is based on a modified Lanczos process and the Fast Fourier Transform.

## 1. Introduction

Structured matrices play an important role in signal processing. A common occurring structure is the Hankel form:

$$H = \begin{pmatrix} h_1 & h_2 & \cdots & h_{n-1} & h_n \\ h_2 & h_3 & \cdots & h_n & h_{n+1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ h_{n-1} & h_n & \cdots & h_{2n-3} & h_{2n-2} \\ h_n & h_{n+1} & \cdots & h_{2n-2} & h_{2n-1} \end{pmatrix},$$

where all elements along the same anti-diagonal are identical. There is an extensive literature on inverting Hankel matrices or solving linear equations with a Hankel structure; for a good list of references, see Golub-Van Loan[4]. The work is more limited on efficient eigenvalue computation for Hankel matrices; some examples are Cybenko-Van Loan[2] Luk-Qiao[6] and Trench[7].

In this paper, we present an  $O(n^2 \log n)$  algorithm for finding all the singular values of an  $n$ -by- $n$  complex Hankel matrix  $H$ . Specifically, we present an algorithm for computing the Takagi decomposition (Horn-Johnson[5]):

$$(1) \quad H = Q\Sigma Q^T,$$

where  $Q$  is unitary and  $\Sigma$  is diagonal with the singular values of  $H$  on its diagonal. An  $O(n^3)$  algorithm for computing the Takagi decomposition of a general complex-symmetric matrix is given by Bunse-Gerstner and Gragg [1]. Their algorithm consists of two stages. First, a complex-symmetric matrix is reduced to a tridiagonal form using Householder transformations. Second, a complex-symmetric tridiagonal matrix is diagonalized by the QR method. They state [1, page 42]:

---

2000 *Mathematics Subject Classification.* Primary 15A23, 65F23.

*Key words and phrases.* Complex Hankel matrix, Takagi decomposition, Lanczos process.

In many applications the symmetric matrix is a Hankel matrix and it is an open question whether there is a method for the symmetric SVD computation which can exploit this by far more special structure to reduce the number of operations and the storage requirement.

Nonetheless, it is remarked [1, page 51] that an  $n$ -by- $n$  Hankel matrix  $H$  “can in principle be tridiagonalized by a Takagi-Lanczos process in  $O(n^2 \log_2 n)$  operations, using FFT’s to compute  $H\mathbf{x}$ .” However, no further details are presented. We use the Lanczos and FFT procedures in this paper, before we became aware of the work by Bunse-Gerstner and Gragg [1].

Our paper is organized as follows. A Lanczos tridiagonalization of a Hankel matrix is described in Section 2, and a two-by-two Takagi decomposition in Section 3. A QR method for the diagonalization of a tridiagonal complex-symmetric matrix is given in Section 4, followed by an overall algorithm and an illustrative numerical example in Section 5.

**Notations.** We use the “bar” symbol to denote a complex conjugate: for example,  $\bar{M}$ ,  $\bar{\mathbf{v}}$  and  $\bar{\alpha}$  denote the complex conjugates of a matrix  $M$ , a vector  $\mathbf{v}$  and a scalar  $\alpha$ , respectively.

## 2. Lanczos Tridiagonalization

As in the standard SVD computation, we first reduce the Hankel matrix to a simpler form, specifically a complex-symmetric tridiagonal matrix.

Consider a Lanczos-like algorithm to reduce  $H$  to an upper Hessenberg form:

$$(2) \quad H\bar{Q} = QK,$$

where the matrix  $Q$  is unitary and the matrix  $K$  is upper Hessenberg. Let

$$Q = (\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_n)$$

and

$$K = (\kappa_{ij}).$$

Writing out equation (2) in column form, we obtain a recurrence formula to compute  $Q$  and  $K$ :

$$(3) \quad \kappa_{l+1,l}\mathbf{q}_{l+1} = H\bar{\mathbf{q}}_l - \kappa_{l,l}\mathbf{q}_l - \kappa_{l-1,l}\mathbf{q}_{l-1} - \cdots - \kappa_{1,l}\mathbf{q}_1.$$

Since

$$K = Q^H H\bar{Q},$$

the matrix  $K$  is complex-symmetric. From symmetry, we deduce that  $K$  is tridiagonal:

$$(4) \quad K = \begin{pmatrix} \alpha_1 & \beta_1 & & & & 0 \\ \beta_1 & \alpha_2 & \beta_2 & & & \\ & \beta_2 & \alpha_3 & \beta_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & \beta_{n-2} & \alpha_{n-1} & \beta_{n-1} \\ 0 & & & & \beta_{n-1} & \alpha_n \end{pmatrix}.$$

The relation (3) thus reduces to a three term recursion:

$$(5) \quad \beta_l\mathbf{q}_{l+1} = H\bar{\mathbf{q}}_l - \alpha_l\mathbf{q}_l - \beta_{l-1}\mathbf{q}_{l-1}.$$

As  $\mathbf{q}_i^H \mathbf{q}_j = \delta_{ij}$ , we get from (5) that  $\alpha_l = \mathbf{q}_l^H H \bar{\mathbf{q}}_l$ . Let

$$\mathbf{r}_l = H \bar{\mathbf{q}}_l - \alpha_l \mathbf{q}_l - \beta_{l-1} \mathbf{q}_{l-1}.$$

Then (5) shows that  $\beta_l$  is the 2-norm of  $\mathbf{r}_l$  and  $\mathbf{q}_{l+1}$  is the normalized  $\mathbf{r}_l$ , i.e.,

$$\beta_l = \sqrt{\mathbf{r}_l^H \mathbf{r}_l}$$

and

$$\mathbf{q}_{l+1} = \mathbf{r}_l / \beta_l.$$

The triangularization procedure is summarized in the following algorithm.

**ALGORITHM 1** (Lanczos Tridiagonalization). *Given an  $n$ -by- $n$  Hankel matrix  $H$ , this algorithm computes a unitary matrix  $Q$  such that  $H = Q K Q^T$ , where  $K$  is complex-symmetric and tridiagonal as shown in (4).*

```

Initialize  $\mathbf{q}_1$  such that  $\|\mathbf{q}_1\|_2 = 1$ ;
Set  $\mathbf{r}_0 = \mathbf{q}_1$ ;  $\beta_0 = 1$ ;  $\mathbf{q}_0 = \mathbf{0}$ ;  $l = 0$ ;
while ( $\beta_l \neq 0$ )
     $\mathbf{q}_{l+1} = \mathbf{r}_l / \beta_l$ ;
     $l \leftarrow l + 1$ ;
     $\alpha_l = \mathbf{q}_l^H H \bar{\mathbf{q}}_l$ ;
     $\mathbf{r}_l = H \bar{\mathbf{q}}_l - \alpha_l \mathbf{q}_l - \beta_{l-1} \mathbf{q}_{l-1}$ ;
     $\beta_l = \|\mathbf{r}_l\|_2$ ;
end.       $\square$ 
```

If all  $\beta_l \neq 0$ , then Algorithm 1 runs until  $l = n$ . The dominant cost is the Hankel matrix-vector product  $H \bar{\mathbf{q}}_l$ , for  $l = 1, 2, \dots, n$ . Using an  $O(n \log n)$  Hankel matrix-vector multiplication scheme (cf. Luk-Qiao[6]), we obtain an  $O(n^2 \log n)$  tridiagonalization algorithm.

### 3. Two-by-Two Takagi Decomposition

In this section, we discuss the Takagi decomposition (1) of a 2-by-2 complex-symmetric matrix.

Consider

$$(6) \quad A = \begin{pmatrix} \alpha & \beta \\ \beta & \gamma \end{pmatrix}.$$

We look for a unitary matrix  $Q$  such that

$$(7) \quad Q^H A \bar{Q} = \Sigma,$$

where

$$\Sigma = \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix}$$

and both  $\sigma_1$  and  $\sigma_2$  are nonnegative. Define

$$\text{sign}(x) = \begin{cases} x/|x| & \text{if } x \neq 0, \\ 1 & \text{if } x = 0. \end{cases}$$

If  $\beta = 0$ , then we pick

$$(8) \quad Q = \begin{pmatrix} \sqrt{\text{sign}(\alpha)} & 0 \\ 0 & \sqrt{\text{sign}(\gamma)} \end{pmatrix},$$

which gives

$$Q^H A \bar{Q} = \begin{pmatrix} |\alpha| & 0 \\ 0 & |\gamma| \end{pmatrix}.$$

So we assume  $\beta \neq 0$  from here on. The product  $A\bar{A}$  ( $= AA^H$ ) is Hermitian and nonnegative definite. Indeed,

$$A\bar{A} = \begin{pmatrix} |\alpha|^2 + |\beta|^2 & \alpha\bar{\beta} + \beta\bar{\gamma} \\ \bar{\alpha}\beta + \bar{\beta}\gamma & |\beta|^2 + |\gamma|^2 \end{pmatrix}.$$

We can find an eigenvalue decomposition:

$$(9) \quad V^H (A\bar{A}) V = D^2,$$

where the matrix  $V$  is unitary and the matrix  $D$  nonnegative diagonal. Let

$$V = (\mathbf{v}_1, \mathbf{v}_2)$$

and

$$D = \begin{pmatrix} d_1 & 0 \\ 0 & d_2 \end{pmatrix}.$$

The Takagi decomposition implies that  $A\bar{Q} = Q\Sigma$ . So we look for a normalized vector  $\mathbf{q}$  such that

$$(10) \quad A\bar{\mathbf{q}} = \sigma_i \mathbf{q}, \quad \sigma_i \geq 0.$$

from which we get

$$A\bar{A}\mathbf{q} = \sigma_i A\bar{\mathbf{q}} = \sigma_i^2 \mathbf{q}.$$

Hence  $\mathbf{q}$  is an eigenvector of  $A\bar{A}$  and  $\sigma_i^2$  is the corresponding eigenvalue. Thus,  $\sigma_i^2$  equals either  $d_1^2$  or  $d_2^2$ . We have two cases depending on whether the eigenvalues  $d_1^2$  and  $d_2^2$  are distinct or identical.

First, we assume that  $d_1 \neq d_2$ . Since the eigenvalues of  $A\bar{A}$  are distinct, the eigenvectors are uniquely defined up to a scalar. Thus,  $\mathbf{q}$  is a scalar multiple of either  $\mathbf{v}_1$  or  $\mathbf{v}_2$  and  $\sigma_i$  is either  $d_1$  or  $d_2$ , respectively. Let

$$(11) \quad A\bar{\mathbf{v}}_1 = \xi d_1 \mathbf{v}_1$$

for some scalar  $\xi$  such that  $|\xi| = 1$ . We define

$$(12) \quad \mathbf{q} \equiv \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} = \sqrt{\text{sign}(\xi)} \mathbf{v}_1.$$

Then

$$A\bar{\mathbf{q}} = \sqrt{\text{sign}(\xi)} \cdot A\bar{\mathbf{v}}_1 = \sqrt{\text{sign}(\xi)} \cdot \xi d_1 \mathbf{v}_1 = |\xi| d_1 \mathbf{q} = d_1 \mathbf{q},$$

as desired. We can get  $\xi$  from (11):

$$(13) \quad \xi = \text{sign}(\mathbf{v}_1^H A\bar{\mathbf{v}}_1).$$

Second, we assume that the eigenvalues are identical, i.e.,  $d_1 = d_2$ . Then  $A\bar{A} = d_1^2 I$  and any vector is an eigenvector. Pick  $\mathbf{v}_1 = \mathbf{e}_1$ . Then  $A\bar{\mathbf{v}}_1 \neq \eta \mathbf{v}_1$  for any scalar  $\eta$ , since  $\beta \neq 0$ . We propose

$$\mathbf{u} = A\bar{\mathbf{e}}_1 + d_1 \mathbf{e}_1 = \begin{pmatrix} \alpha + d_1 \\ \beta \end{pmatrix}.$$

Then

$$A\bar{\mathbf{u}} = A\bar{A}\mathbf{e}_1 + d_1 A\bar{\mathbf{e}}_1 = d_1^2 \mathbf{e}_1 + d_1 A\bar{\mathbf{e}}_1 = d_1 \mathbf{u}.$$

We choose  $\mathbf{q}$  as a normalized  $\mathbf{u}$  with  $q_1 \geq 0$ . Let

$$(14) \quad \mathbf{q} \equiv \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} = \frac{1}{\sqrt{|\alpha + d_1|^2 + |\beta|^2}} \begin{pmatrix} |\alpha + d_1| \\ \beta(\bar{\alpha} + \bar{d}_1)/|\alpha + d_1| \end{pmatrix}.$$

Given  $\mathbf{q}$  in either (12) or (14), we can construct a unitary matrix  $Q$  by

$$(15) \quad Q = \begin{pmatrix} q_1 & -\bar{q}_2 \\ q_2 & \bar{q}_1 \end{pmatrix}.$$

Thus, the product

$$Q^H A \bar{Q} = \begin{pmatrix} d_1 & \times \\ 0 & \times \end{pmatrix}$$

is upper triangular. As the matrix  $Q^H A \bar{Q}$  is symmetric, it is therefore diagonal. In fact, we can readily show that

$$Q^H A \bar{Q} = \begin{pmatrix} d_1 & 0 \\ 0 & d_2 \end{pmatrix},$$

assuming that the second column of  $Q$  has been scaled so that  $d_2 \geq 0$ .

**ALGORITHM 2** (2-by-2 Takagi Decomposition). *Given a 2-by-2 complex symmetric  $A$  in (6), this algorithm computes a 2-by-2 unitary matrix  $Q$  so that (7) is satisfied.*

If  $\beta = 0$  then pick  $Q$  as in (8)

else

Find an eigenvalue decomposition of  $A\bar{A}$ :  $V^H A \bar{A} V = D^2$ ;

If  $d_1 \neq d_2$  then set  $\mathbf{q}$  using (12) ( $\xi$  from (13)) else set  $\mathbf{q}$  using (14);

Construct unitary matrix  $Q$  as in (15)

end. □

#### 4. Diagonalization

This section concerns the Takagi decomposition of an  $n$ -by- $n$  tridiagonal matrix  $K$  of (4):

$$K = Q \Sigma Q^T,$$

where  $Q$  is unitary and  $\Sigma$  is nonnegative diagonal. We apply an implicit QR algorithm; that is, instead of an explicit formation of  $K^H K$ , we apply Householder transformations directly to  $K$ .

Consider the trailing 3-by-3 submatrix  $\Lambda$  of  $K^H K$ . It is given by

$$(16) \quad \Lambda =$$

$$\begin{pmatrix} \lambda_{11} & \bar{\alpha}_{n-2}\beta_{n-2} + \alpha_{n-1}\bar{\beta}_{n-2} & \bar{\beta}_{n-2}\beta_{n-1} \\ \alpha_{n-2}\bar{\beta}_{n-2} + \bar{\alpha}_{n-1}\beta_{n-2} & \lambda_{22} & \bar{\alpha}_{n-1}\beta_{n-1} + \alpha_n\bar{\beta}_{n-1} \\ \beta_{n-1}\beta_{n-2} & \alpha_{n-1}\bar{\beta}_{n-1} + \bar{\alpha}_n\beta_{n-1} & \lambda_{33} \end{pmatrix},$$

where

$$\lambda_{11} = |\beta_{n-3}|^2 + |\alpha_{n-2}|^2 + |\beta_{n-2}|^2,$$

$$\lambda_{22} = |\beta_{n-2}|^2 + |\alpha_{n-1}|^2 + |\beta_{n-1}|^2,$$

$$(17) \quad \lambda_{33} = |\beta_{n-1}|^2 + |\alpha_n|^2.$$

Let  $\lambda$  denote the eigenvalue of  $\Lambda$  of that is the closest to  $\lambda_{33}$ , and let  $J_1$  be the Householder matrix such that

$$J_1^T \mathbf{x} = (\times, 0, 0)^T,$$

where

$$(18) \quad \mathbf{x} = \begin{pmatrix} |\alpha_1|^2 + |\beta_1|^2 - \lambda \\ \alpha_1 \bar{\beta}_1 + \bar{\alpha}_2 \beta_1 \\ \beta_1 \bar{\beta}_2 \end{pmatrix}.$$

We note that the vector  $\mathbf{x}$  consists of the top three elements from the first column of  $K^H K - \lambda I$ . Let us apply

$$\tilde{J}_1 = \begin{pmatrix} J_1^T & 0 \\ 0 & 1 \end{pmatrix}$$

to a block in  $K$  directly:

$$(19) \quad \begin{pmatrix} \alpha_1 & \beta_1 & \gamma_1 & \delta_1 \\ \beta_1 & \alpha_2 & \beta_2 & \gamma_2 \\ \gamma_1 & \beta_2 & \alpha_3 & \beta_3 \\ \delta_1 & \gamma_2 & \beta_3 & \alpha_4 \end{pmatrix} \leftarrow \tilde{J}_1^T \begin{pmatrix} \alpha_1 & \beta_1 & & \\ \beta_1 & \alpha_2 & \beta_2 & \\ \beta_2 & \alpha_3 & \beta_3 & \\ \beta_3 & \alpha_4 & & \end{pmatrix} \tilde{J}_1.$$

We follow with Householder transformations  $J_2, J_3, \dots, J_{n-2}$  to restore the tridiagonal structure of  $K$ , while maintaining symmetry at the same time. To illustrate, we determine a Householder matrix  $J_k$  such that

$$J_k^T (\beta_{k-1}, \gamma_{k-1}, \delta_{k-1})^T = (\times, 0, 0)^T.$$

Let

$$\tilde{J}_k = \begin{pmatrix} 1 & 0 & 0 \\ 0 & J_k^T & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Then

$$(20) \quad \begin{pmatrix} \alpha_{k-1} & \beta_{k-1} & & & \\ \beta_{k-1} & \alpha_k & \beta_k & \gamma_k & \delta_k \\ \beta_k & \alpha_{k+1} & \beta_{k+1} & \gamma_{k+1} & \\ \gamma_k & \beta_{k+1} & \alpha_{k+2} & \beta_{k+2} & \\ \delta_k & \gamma_{k+1} & \beta_{k+2} & \alpha_{k+3} & \end{pmatrix} \leftarrow \tilde{J}_k^T \begin{pmatrix} \alpha_{k-1} & \beta_{k-1} & \gamma_{k-1} & \delta_{k-1} & \\ \beta_{k-1} & \alpha_k & \beta_k & \gamma_k & \\ \gamma_{k-1} & \beta_k & \alpha_{k+1} & \beta_{k+1} & \\ \delta_{k-1} & \gamma_k & \beta_{k+1} & \alpha_{k+2} & \beta_{k+2} \\ & & & \beta_{k+2} & \alpha_{k+3} \end{pmatrix} \tilde{J}_k.$$

Consider the new matrix  $K^{(k)} \leftarrow \hat{J}_k^T \cdots \hat{J}_2^T \hat{J}_1^T K \hat{J}_1 \hat{J}_2 \cdots \hat{J}_k$ , where  $\hat{J}_i$  denotes the appropriate  $n \times n$  Householder matrix that contains the lower-dimension  $\tilde{J}_i$  as a submatrix. The resultant matrix is symmetric and the bulge is chased down by one row and one column. Eventually, the bulge is chased out of the matrix and the final new matrix

$$(21) \quad K \equiv K^{(n-1)} \leftarrow \hat{J}_{n-1}^T \cdots \hat{J}_2^T \hat{J}_1^T K \hat{J}_1 \hat{J}_2 \cdots \hat{J}_{n-1}$$

is symmetric and tridiagonal. As in the standard implicit QR method, the new  $K$  is closer to diagonal. Thus, we have derived an algorithm for one step of a complex-symmetric SVD.

ALGORITHM 3 (One Step of Complex-Symmetric SVD). *Given diagonal vector*

$$\mathbf{a} = (\alpha_1, \alpha_2, \dots, \alpha_n)^T$$

*and subdiagonal vector*

$$\mathbf{b} = (\beta_1, \beta_2, \dots, \beta_{n-1})^T$$

*of a tridiagonal matrix  $K$  of (4), this algorithm overwrites  $\mathbf{a}$  and  $\mathbf{b}$  so that the new matrix  $K$  formed by the new  $\mathbf{a}$  and  $\mathbf{b}$  is the same as the matrix obtained by applying one step of an implicit QR algorithm.*

If  $n = 2$ , apply Algorithm 2 and return;

Compute  $\lambda$ , the eigenvalue of  $\Lambda$  of (16) that is closest to  $\lambda_{33}$  of (17);

for  $k = 1 : n - 2$

if  $k = 1$  then set  $\mathbf{x}$  using (18) else set  $\mathbf{x} = (\beta_{k-1}, \gamma_{k-1}, \delta_{k-1})^T$ ;

Determine a Householder matrix  $J_k$  such that  $J_k^T \mathbf{x} = (\times, 0, 0)^T$ ;

if  $k = 1$  then update block as in (19) else update block as in (20);

end

end

Determine a Householder matrix  $J_{n-1}$  s.t.  $J_{n-1}^T(\beta_{n-2}, \gamma_{n-2})^T = (\times, 0)^T$ ;

Update the last block:

$$\begin{pmatrix} \alpha_{n-2} & \beta_{n-2} & 0 \\ \beta_{n-2} & \alpha_{n-1} & \beta_{n-1} \\ 0 & \beta_{n-1} & \alpha_n \end{pmatrix} \leftarrow \\ \begin{pmatrix} 1 & 0 \\ 0 & J_{n-1}^T \end{pmatrix} \begin{pmatrix} \alpha_{n-2} & \beta_{n-2} & \gamma_{n-2} \\ \beta_{n-2} & \alpha_{n-1} & \beta_{n-1} \\ \gamma_{n-2} & \beta_{n-1} & \alpha_n \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & J_{n-1} \end{pmatrix}. \quad \square$$

The one-step algorithm requires  $O(n)$  flops. Accumulating  $J_i$  requires  $O(n^2)$  flops. The method is used in the singular value computation. Let  $q$  be the largest integer such that

$$\mathbf{b}_{n-q:n} = \mathbf{0},$$

i.e., the subvector  $\mathbf{b}_{n-q:n}$  is the null vector. (Initially, we have  $q = 0$  and  $\beta_n = 0$ .) Also, let  $p$  denote the smallest integer such that the subvector  $\mathbf{b}_{p+1:n-q-1}$  has no zero entries. Then the principal submatrix

$$B = \begin{pmatrix} \alpha_{p+1} & \beta_{p+1} & & & 0 \\ \beta_{p+1} & \alpha_{p+2} & \beta_{p+2} & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \beta_{n-q-1} \\ 0 & & & \beta_{n-q-1} & \alpha_{n-q} \end{pmatrix}$$

has no zeros on its subdiagonal. We apply Algorithm 3 to  $B$ . When some  $\beta_i$  becomes sufficiently small:

$$(22) \quad |\beta_i| \leq c(|\alpha_i| + |\alpha_{i+1}|)u,$$

where  $c$  denotes a small constant and  $u$  the unit roundoff, then we set  $\beta_i$  to zero and update  $p$  and  $q$ . When  $q$  reaches  $n - 1$ ,  $K$  becomes diagonal with the singular values on the diagonal.

**ALGORITHM 4** (Complex-symmetric SVD). *Given the diagonal  $\mathbf{a}$  and subdiagonal  $\mathbf{b}$  of the tridiagonal matrix  $K$  of (4), this algorithm computes the Takagi decomposition  $K = Q\Sigma Q^T$ . The diagonal  $\mathbf{a}$  is overwritten by the singular values.*

```

Initialize  $q = 0$ ,  $\beta_n = 0$ , and  $Q = I$ ;
while  $q < n - 1$ 
  Set all  $\beta_i$  satisfying (22) to zero;
  Update  $q$  so that  $\mathbf{b}_{n-q:n} = \mathbf{0}$  and  $\beta_{n-q-1} \neq 0$ ;
  Find the smallest  $p$  so that  $\mathbf{b}_{p+1:n-q-1}$  has no zero entries;
  If  $q < n - 1$ 
    Apply Algorithm 3 to the complex-symmetric and
    tridiagonal matrix whose diagonal and subdiagonal
    are  $\mathbf{a}_{p+1:n-q}$  and  $\mathbf{b}_{p+1:n-q-1}$ , respectively;
    Update  $Q$ ;
  end
end.   □

```

As stated before, Algorithm 3 requires  $O(n)$  flops without accumulating the Householder transformations. Thus, Algorithm 4 uses  $O(n^2)$  flops without explicitly forming  $Q$ .

## 5. Overall Procedure

We conclude our paper with an overall singular value procedure and an illustrative numerical example.

**ALGORITHM 5** (Fast Hankel Singular Value Algorithm). *Given a complex Hankel matrix  $H$ , this algorithm computes all its singular values.*

1. Apply Algorithm 1 to  $H$  to obtain a symmetric and tridiagonal  $K$ ;
2. Apply Algorithm 4 to calculate the singular values of  $K$ . □

The major cost of this algorithm is the tridiagonalization procedure and the dominant cost of the Lanczos tridiagonalization is matrix-vector multiplication. The  $O(n \log n)$  Hankel matrix-vector multiplication scheme is faster than general  $O(n^2)$  matrix-vector multiplication when  $n \geq 16$  (cf. Luk-Qiao [6]). We expect our proposed Hankel SVD algorithm to be faster than a general SVD algorithm for a very small value of  $n$ .

We know that, in a straightforward implementation of Lanczos procedure, the orthogonality of the vectors  $\mathbf{q}_l$  in Algorithm 1 deteriorates as the size of  $H$  increases. Reorthogonalization is necessary for a practical Lanczos method. Efficient and practical reorthogonalization techniques are available, see [3, §7.5] and references there. They achieve the orthogonality of  $\mathbf{q}_l$  nearly as good as complete reorthogonalization with just a little extra work.

**Example.** Suppose that the first column and the last row of  $H$  are respectively

$$\left( \begin{array}{c} 0.9501 + 0.7621i \\ 0.2311 + 0.4565i \\ 0.6068 + 0.0185i \\ 0.4860 + 0.8214i \\ 0.8913 + 0.4447i \end{array} \right) \quad \text{and} \quad \left( \begin{array}{c} 0.8913 + 0.4447i \\ 0.7919 + 0.9355i \\ 0.9218 + 0.9169i \\ 0.7382 + 0.4103i \\ 0.1763 + 0.8937i \end{array} \right).$$

After tridiagonalization, the diagonal and subdiagonal of  $K$  are respectively

$$\begin{pmatrix} 3.4438 + 3.0893i \\ 0.1558 + 0.1970i \\ 0.1729 + 0.0537i \\ 0.3771 + 0.0265i \\ -0.7437 + 0.4832i \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 0.5400 \\ 0.6584 \\ 0.5859 \\ 0.4940 \end{pmatrix}.$$

The following table presents the four subdiagonal elements of  $K$  during the execution of Algorithm 4.

Iter.	$\beta_1$	$\beta_2$	$\beta_3$	$\beta_4$
1	$-0.017 + 0.000i$	$0.491 - 0.087i$	$-0.531 + 0.115i$	$-0.233 - 0.041i$
2	$10^{-4}$	$0.389 - 0.071i$	$0.215 - 0.050i$	$10^{-3}$
3	$10^{-5}$	$0.318 - 0.058i$	$-0.072 + 0.017i$	$10^{-8}$
4	$10^{-6}$	$0.253 - 0.046i$	$0.024 - 0.006i$	converged
5	$10^{-8}$	$0.209 - 0.038i$	converged	
6	$10^{-9}$	$10^{-15}$		
7	$10^{-11}$	converged		
8	converged			

The computed singular values are  $\{ 4.6899, 1.1819, 1.0673, 0.62109, 0.37028 \}$ . Assuming that the MATLAB function `svd()` is fully accurate, we find the errors in the singular values computed by Algorithm 5 to be  $10^{-15}$ .

**Acknowledgement.** The authors thank Professor Angelika Bunse-Gerstner for her valuable comment.

## References

- [1] A. Bunse-Gerstner and W. B. Gragg, “Singular value decompositions of complex symmetric matrices,” *J. Comput. Appl. Math.*, 21, 1988, pp. 41–54.
- [2] G. Cybenko and C.F. Van Loan, “Computing the minimum eigenvalue of a symmetric positive definite Toeplitz matrix,” *SIAM J. Sci. Statist. Comput.*, 7, 1986, pp. 123–131.
- [3] J. W. Demmel, *Applied Numerical Linear Algebra*, Society for Industrial and Applied Mathematics, 1997.
- [4] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd Ed., The Johns Hopkins University Press, Baltimore, MD, 1996.
- [5] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, 1985.
- [6] F. T. Luk and S. Qiao, “A fast eigenvalue algorithm for Hankel matrices,” *Lin. Alg. Applics.*, 316, 2000, pp. 171–182.
- [7] W. F Trench, “Numerical solution of the eigenvalue problem for Hermitian Toeplitz matrices,” *SIAM J. Matrix Anal. Appl.*, Vol. 10, No. 2, 1989, pp. 135–146.

DEPT OF COMPUTER SCIENCE, RENSSELAER POLYTECHNIC INSTITUTE, TROY, NY 12180, USA  
*E-mail address:* luk@cs.rpi.edu

DEPT OF COMPUTING AND SOFTWARE, McMMASTER UNIV., HAMILTON, ONT L8S 4L7, CANADA  
*E-mail address:* qiao@mcmaster.ca

*This page intentionally left blank*

## A modified companion matrix method based on Newton polynomials

D. Calvetti, L. Reichel, and F. Sgallari

**ABSTRACT.** One of the most popular methods for computing the zeros of a polynomial in power form is to determine the eigenvalues of the associated companion matrix. This approach, which commonly is referred to as the companion matrix method, however, can yield poor accuracy. We demonstrate that when approximations of the zeros are available, their accuracy often can be improved by computing the eigenvalues of a modified companion matrix associated with the polynomial expressed in a basis of Newton polynomials defined by the available approximations of the zeros. The latter approximations can be determined in many ways, e.g., by the companion matrix method.

### 1. Introduction

The computation of the zeros  $z_1, z_2, \dots, z_n$  of a polynomial

$$(1) \quad \psi_n(z) = z^n + \alpha_{n-1}z^{n-1} + \dots + \alpha_1z + \alpha_0, \quad \alpha_j \in \mathbb{C},$$

is a fundamental task in scientific computation that arises in many problems in science and engineering. Many numerical methods have been proposed and analyzed for this problem; see, e.g., Henrici [5, Chapter 6] and Stoer and Bulirsch [12, Chapter 5] as well as the extensive bibliography provided by McNamee [9]. Among the most popular numerical schemes for polynomials of small to moderate degrees are the Jenkins-Traub method [6] and the companion matrix method. The latter method computes the zeros of the polynomial (1) as the eigenvalues of the companion matrix

$$(2) \quad C_n = \begin{bmatrix} 0 & & \cdots & 0 & -\alpha_0 \\ 1 & 0 & & 0 & -\alpha_1 \\ & 1 & 0 & \cdots & 0 & -\alpha_2 \\ & & & & \vdots & \vdots \\ & & & \ddots & \ddots & \\ & & & & 1 & 0 & -\alpha_{n-2} \\ 0 & & & & & 1 & -\alpha_{n-1} \end{bmatrix} \in \mathbb{C}^{n \times n}$$

---

2000 *Mathematics Subject Classification.* Primary 65H05, 65F15, 26C10.

*Key words and phrases.* Companion matrix, Newton polynomial, zero-finder.

Research supported in part by NSF grants DMS 0107841 and 0107858.

by the QR-algorithm after balancing; see Edelman and Murakami [2] and Moler [10] for discussions. This method is used in the popular software package MATLAB [8] when calling the function `roots`. Goedecker [3] compared several implementations of the Jenkins-Traub method with the companion matrix method with regard to both accuracy and execution time for polynomials of small to moderate degree and found the latter approach to be competitive. We will throughout this paper refer to the companion matrix method as the CB method (where the letter C refers to the companion matrix and B to balancing).

However, it is fairly easy to find polynomials for which the CB method gives poor accuracy. Two modifications of the CB methods, referred to as the CBS and SHB methods, that for some polynomials can yield higher accuracy than the CB method were proposed in [1]. The CBS method first translates and scales the polynomial (1) to obtain a new polynomial

$$(3) \quad \hat{\psi}_n(\hat{z}) = \hat{z}^n + \hat{\alpha}_{n-2}\hat{z}^{n-2} + \dots + \hat{\alpha}_1\hat{z} + \hat{\alpha}_0, \quad \hat{\alpha}_j \in \mathbb{C},$$

with all zeros  $\hat{z}_1, \hat{z}_2, \dots, \hat{z}_n$  inside the unit disk and at least one zero fairly close to the unit circle, and such that  $\sum_{j=1}^n \hat{z}_j = 0$ . Thus, the zeros  $z_j$  of  $\psi_n$  and  $\hat{z}_j$  of  $\hat{\psi}_n$  are related according to

$$(4) \quad z_j = \sigma \hat{z}_j - \rho, \quad 1 \leq j \leq n,$$

where the translation is given by  $\rho = \alpha_{n-1}/n$  and the scaling factor  $\sigma$  is determined by using an inequality due to Ostrowski and the Schur-Cohn test: see [1] for details. Having determined the coefficients  $\hat{\alpha}_j$  of  $\hat{\psi}_n$ , we compute the zeros  $\hat{z}_j$  as the eigenvalues of the companion matrix associated with the polynomial  $\hat{\psi}_n$ . The desired zeros  $z_j$  are then determined by (4).

The SHB method is based on the observation that the polynomial (3) is a member of a family of monic Szegő polynomials  $\phi_0, \phi_1, \dots, \phi_n, \dots$ , where  $\hat{\psi}_n(\hat{z}) = \phi_n(\hat{z})$ . Szegő polynomials are orthogonal with respect to an inner product on the unit circle; see [4] for some of their properties and applications. The polynomial (3) can be expressed in terms of Szegő polynomials of lower degree

$$\hat{\psi}_n(\hat{z}) = \hat{z}\phi_{n-1}(\hat{z}) + \hat{\beta}_{n-1}\phi_{n-1}(\hat{z}) + \hat{\beta}_{n-2}\phi_{n-2}(\hat{z}) + \dots + \hat{\beta}_1\phi_1(\hat{z}) + \hat{\beta}_0.$$

This expression and the recursion formulas for the Szegő polynomials define a Hessenberg matrix, whose eigenvalues are the zeros  $\hat{z}_j$  of  $\hat{\psi}_n$ . The eigenvalues are computed by the QR-algorithm and the desired zeros  $z_j$  of  $\psi_n$  are obtained from (4).

This paper presents a new approach to improve the accuracy of available approximations  $\hat{z}'_1, \hat{z}'_2, \dots, \hat{z}'_n$  of the zeros of  $\hat{\psi}_n$ . These approximations may, for instance, have been computed by the CB, CBS or SHB methods. Let  $z'_1, z'_2, \dots, z'_n$  denote the associated approximations of the zeros of  $\psi_n$ , i.e.,  $z'_j = \sigma \hat{z}'_j - \rho$ . Express the polynomial  $\hat{\psi}_n$  in terms of a basis of Newton polynomials determined by the nodes  $\hat{z}'_1, \hat{z}'_2, \dots, \hat{z}'_n$ , i.e.,

$$(5) \quad \hat{\psi}_n(\hat{z}) = \hat{\gamma}_0 + \hat{\gamma}_1(\hat{z} - \hat{z}'_1) + \hat{\gamma}_2(\hat{z} - \hat{z}'_1)(\hat{z} - \hat{z}'_2) + \dots + \hat{\gamma}_n \prod_{j=1}^n (\hat{z} - \hat{z}'_j)$$

with  $\hat{\gamma}_n = 1$ . We define a modified companion matrix associated with the Newton form (5). The eigenvalues of this modified companion matrix are the zeros of  $\hat{\psi}_n$ . We compute the eigenvalues by the QR-algorithm after balancing of the matrix.

This defines the modified companion matrix (MCB) method. Let  $\hat{z}_1'', \hat{z}_2'', \dots, \hat{z}_n''$  denote the zeros computed by the MCB method, and let  $z_1'', z_2'', \dots, z_n''$  denote the associated approximations of the zeros of  $\psi_n$ , i.e.,  $z_j'' = \sigma \hat{z}_j'' - \rho$ . Then, typically,  $z_j''$  is a better approximation of  $z_j$  than  $\hat{z}_j'$ .

Details of the MCB method are presented in Section 2, computed examples are discussed in Section 3, and concluding remarks can be found in Section 4.

## 2. The modified companion matrix method

Let  $\hat{z}_1', \hat{z}_2', \dots, \hat{z}_n'$  be points in the complex plane. In the computations presented in Section 3, the  $\hat{z}_j'$  are computed approximations of zeros of the polynomial (3), however, for the discussion of the present section they may be arbitrary points in the finite complex plane.

We compute the coefficients  $\hat{\gamma}_j$  of the representation (5) of the polynomial  $\hat{\psi}_n$  in Newton form, given the representation (3) in power form and the nodes  $\hat{z}_j'$ . The computations are described by the following algorithm. The algorithm can be used to transform a general polynomial of degree  $n$  with leading coefficient  $\hat{\alpha}_n$  to a polynomial in Newton form. In particular, the algorithm does not use that  $\hat{\alpha}_n = 1$  and  $\hat{\alpha}_{n-1} = 0$ .

ALGORITHM 2.1 (Transformation from power basis to Newton basis).

**Input:**  $n, \{\hat{\alpha}_k\}_{k=0}^n, \{\hat{z}_k'\}_{k=1}^n$ ;

**Output:**  $\{\hat{\gamma}_k\}_{k=0}^n$ ;

$\hat{\gamma}_n := \hat{\alpha}_n$ ;

**for**  $j := 1, 2, \dots, n$  **do**

$\hat{\gamma}_{n-j} := \hat{\alpha}_{n-j}$ ;

**for**  $k := 1, 2, \dots, j$  **do**

$\hat{\gamma}_{n-k} := \hat{\gamma}_{n-k} + \hat{\gamma}_{n-k+1} \hat{z}'_{j-k+1}$ ;

**end**  $k$ ;

**end**  $j$ ;

□

The computations of Algorithm 2.1 require  $\mathcal{O}(n^2)$  arithmetic floating point operations.

In order to avoid large intermediate quantities, which may cause loss of accuracy during the computation with Algorithm 2.1, we Leja order the nodes  $\hat{z}_j'$  before application of the algorithm. This ordering is defined by the following algorithm. The set  $\mathbb{K}$  in the algorithm is the set  $\{\hat{z}_j'\}_{j=1}^n$  of nodes, in arbitrary order, to be used in the Newton form. The algorithm assumes the nodes to be distinct. This restriction can easily be removed, see below.

ALGORITHM 2.2 (Leja ordering of nodes).

**Input:** Set  $\mathbb{K}$  of  $n$  distinct nodes;

**Output:** Leja ordered nodes  $\hat{z}_1', \hat{z}_2', \dots, \hat{z}_n'$ ;

Let  $\hat{z}_1'$  be a node in  $\mathbb{K}$ , such that  $\hat{z}_1' = \max_{\hat{z} \in \mathbb{K}} |\hat{z}|$ ;

**for**  $j := 2, 3, \dots, n$  **do**

    Determine  $\hat{z}_j' \in \mathbb{K}$ , so that

$$\prod_{k=1}^{j-1} |\hat{z}_j' - \hat{z}_k'| = \max_{\hat{z} \in \mathbb{K}_{\ell}} \prod_{k=1}^{j-1} |\hat{z} - \hat{z}_k'| ;$$

**end**  $j$ ;

□

We refer to the ordering determined by Algorithm 2.2 as ‘‘Leja ordering,’’ because when  $\mathbb{K}$  is a compact continuum in the complex plane, the points determined by the algorithm are known as Leja points; see Leja [7] for a definition of Leja points and some of their properties. We have found experimentally that Leja ordering of the nodes before application of Algorithm 2.1 generally avoids that intermediate values of the coefficients  $\hat{\gamma}_{n-k}$  computed during the execution of the algorithm are of large magnitude when the coefficients of the polynomial (3) are not. Large intermediate values may reduce the accuracy in the computed coefficients of the polynomial (3). Algorithm 2.2 can be implemented to require only  $\mathcal{O}(n^2)$  arithmetic floating point operations.

When some nodes in the set  $\mathbb{K}$  in Algorithm 2.2 are not distinct, then these nodes are ordered consecutively. Moreover, when the coefficients  $\alpha_j$  of the polynomial (1) are real, the nodes  $\hat{z}'_j$  are real or appear in complex conjugate pairs. We order distinct complex conjugate nodes consecutively.

Having determined the coefficients  $\hat{\gamma}_j$  of the Newton form (5) by application of Algorithms 2.2 and 2.1, we form the modified companion matrix associated with the Newton form. It is given by

$$(6) \quad N_n = \begin{bmatrix} \hat{z}'_1 & & \cdots & 0 & -\hat{\gamma}_0 \\ 1 & \hat{z}'_2 & \cdots & 0 & -\hat{\gamma}_1 \\ 1 & \hat{z}'_3 & \cdots & 0 & -\hat{\gamma}_2 \\ & & & \vdots & \vdots \\ & & & \ddots & \ddots \\ 0 & & & 1 & \hat{z}'_{n-1} & -\hat{\gamma}_{n-2} \\ & & & & 1 & \hat{z}'_n - \hat{\gamma}_{n-1} \end{bmatrix} \in \mathbb{C}^{n \times n}.$$

We compute the eigenvalues of the matrix  $N_n$  by the QR-algorithm after balancing and denote the computed eigenvalues by  $\hat{z}''_j$ ,  $1 \leq j \leq n$ . This defines the MCB method.

In our application of the MCB method, the nodes  $\hat{z}'_j$  are approximations of zeros of  $\hat{\psi}_n$  computed by the CB, CBS or SHB methods. Typically, the zeros  $\hat{z}''_j$  computed by the MCB method are better approximations of the zeros  $\hat{z}_j$  of  $\hat{\psi}_n$  than the nodes  $\hat{z}'_j$ . We refer to the combined method that first determines approximations  $\hat{z}'_j$  of the zeros  $\hat{\psi}_n$  by the CB method, and then improves their accuracy by the MCB method as the CB+MCB method. The CBS+MCB and SHB+MCB methods are defined analogously.

Note that if the nodes  $\hat{z}'_j$  are the exact zeros of the polynomial (3), then the coefficients  $\hat{\gamma}_j$ , when evaluated exactly, all vanish, and the matrix  $N_n$  is lower bidiagonal. Its eigenvalues are the exact zeros of the polynomial (3). If the nodes  $\hat{z}'_j$  are close to the exact zeros, then the coefficients  $\hat{\gamma}_j$  are of small magnitude and the matrix  $N_n$  is close to a lower bidiagonal matrix.

We may apply the MCB method recursively by using the approximations  $\hat{z}'_j$  of the zeros obtained by the MCB method to define a new Newton basis and then applying the MCB method to compute new approximations  $\hat{z}'''_j$  of the zeros. For instance, we may define the CB+MCB+MCB method in this manner. However, computational experience suggests that the latter method typically does not yield higher accuracy of the computed zeros than the CB+MCB method. Our experience with the CBS+MCB+MCB and SHB+MCB+MCB methods is similar.

TABLE 1. Comparison of zero finders applied to polynomials of degree 30 with zeros uniformly distributed in the square (7) with no pair of zeros closer than 0.2.

Method	Average Error	Smallest Error
SHB+MCB	$5.8 \cdot 10^{-5}$	97%
SHB	$6.2 \cdot 10^{-3}$	3%
CBS+MCB	$5.6 \cdot 10^{-5}$	97%
CBS	$1.8 \cdot 10^{-2}$	3%
CB+MCB	$1.6 \cdot 10^{-4}$	99%
CB	$4.8 \cdot 10^{-3}$	1%

### 3. Computed examples

This section presents computed examples that illustrate the performance of the CB+MCB, CBS+MCB and SHB+MCB methods. The computer programs used were all written in FORTRAN 77, and the numerical experiments were carried out on an HP9000/770 workstation in single-precision arithmetic, i.e., with approximately 7 significant decimal digits of accuracy. The eigenvalue problems were solved by single-precision subroutines from EISPACK [11].

In our experiments, we provided a set of  $n$  real or complex conjugate zeros of the polynomial  $\psi_n$ , defined by (1), and computed the coefficients  $\alpha_j$  of the power basis representation by a recursion formula. The zeros were Leja ordered by Algorithm 2.2 in order to avoid unnecessary loss of accuracy; see the discussion in Section 2. The zeros were uniformly distributed in the square

$$(7) \quad \mathbb{S} = \{z \in \mathbb{C} : -1 \leq \operatorname{Re}(z) \leq 1, -1 \leq \operatorname{Im}(z) \leq 1\}$$

or in the unit disk

$$(8) \quad \mathbb{D} = \{z \in \mathbb{C} : |z| \leq 1\}.$$

Since the zeros of  $\psi_n$  are real or appear in complex conjugate pairs, the matrices whose eigenvalues need to be computed in the CB, CBS and SHB methods are real and of upper Hessenberg form. We computed the eigenvalues of these matrices using the EISPACK subroutines `balanc` and `hqr`. The former implements a balancing scheme, the latter the QR algorithm. The modified companion matrix (6) has complex entries whenever  $\psi_n$  has non-real zeros. Its eigenvalues were computed by the EISPACK subroutine `comqr` after balancing by the subroutine `cbal`.

We measure the accuracy of the approximate zeros computed by the different methods by determining the distance between the set of exact zeros  $\mathbb{Z}_n = \{z_j\}_{j=1}^n$  and the set of computed eigenvalues, say,  $\mathbb{Z}'_n = \{z'_j\}_{j=1}^n$ . The distance is defined by

$$(9) \quad \operatorname{distance}(\mathbb{Z}_n, \mathbb{Z}'_n) = \max\{\max_{z'' \in \mathbb{Z}'_n} \|z'' - \mathbb{Z}_n\|, \max_{z \in \mathbb{Z}_n} \|z - \mathbb{Z}'_n\|\},$$

where  $\|z'' - \mathbb{Z}_n\| = \min_{z \in \mathbb{Z}_n} |z'' - z|$ . We refer to  $\operatorname{distance}(\mathbb{Z}_n, \mathbb{Z}'_n)$  as the error in the approximate zeros.

Example 3.1. We consider polynomials of degree 30, whose zeros are uniformly distributed in the square (7). The zeros are either real or appear in complex conjugate pairs, and the distance between each pair of zeros is required to be at least

TABLE 2. Comparison of zero finders applied to polynomials of degree 30 with zeros uniformly distributed in the square (7) with no pair of zeros closer than 0.2.

Error Quotient	Min	Max
error(SHB+MCB)/error(SHB)	$3.2 \cdot 10^{-4}$	4.6
error(CBS+MCB)/error(CBS)	$1.8 \cdot 10^{-4}$	1.0
error(CB+MCB)/error(CB)	$9.9 \cdot 10^{-4}$	1.7

TABLE 3. Comparison of zero finders applied to polynomials of degree 30 with zeros uniformly distributed in the square (7) with no pair of zeros closer than 0.1.

Method	Average Error	Smallest Error
SHB+MCB	$4.2 \cdot 10^{-3}$	93%
SHB	$2.4 \cdot 10^{-2}$	7%
CBS+MCB	$4.9 \cdot 10^{-3}$	98%
CBS	$4.6 \cdot 10^{-2}$	2%
CB+MCB	$1.8 \cdot 10^{-3}$	98%
CB	$1.2 \cdot 10^{-2}$	1%

0.2. The latter requirement is imposed in order to avoid severe ill-conditioning of the eigenvalue problems. For severely ill-conditioned eigenvalue problems the improvement in accuracy achieved by the MCB method is typically insignificant.

Table 1 shows the average error obtained by the different zero finders when applied to 100 polynomials of degree 30 generated as described above. The error in the polynomial zeros is defined according to (9). The SHB+MCB and CBS+MCB methods can be seen to give smaller errors than the other methods.

The MCB method reduces the error in the computed zeros of almost all polynomials. The last column of Table 1 shows for how many polynomials (out of 100) the MCB method reduces the error. For instance, the SHB+MCB method yields a smaller error than the SHB method for 97 of the 100 generated polynomials.

It is interesting to see how much the MCB method decreases or increases the error in available approximations of zeros. We therefore evaluate, for each polynomial, the quotient of the error in the computed zeros by the SHB and the SHB+MCB methods. The maximum and minimum of these quotients over the 100 polynomials in our experiment are displayed in Table 2. The table also shows the corresponding quotients associated with the CBS, CBS+MCB, CB and CB+MCB methods.

We remark that the errors reported in Tables 1-2 are for 100 consecutive polynomials determined by a random number generator. The entries of the tables vary somewhat with the initial seed of the generator, however, in all our experiments the SHB+MCB and CBS+MCB methods gave the smallest average errors in the computed zeros.  $\square$

Example 3.2. This example differs from Example 3.1 only in that the zeros are allowed to be closer. More precisely, the real or complex conjugate zeros of polynomials of degree 30 are uniformly distributed in the square (7), and the distance between each pair of zeros is required to be at least 0.1. Reducing the distance

TABLE 4. Comparison of zero finders applied to polynomials of degree 30 with zeros uniformly distributed in the square (7) with no pair of zeros closer than 0.1.

Error Quotient	Min	Max
error(SHB+MCB)/error(SHB)	$2.8 \cdot 10^{-4}$	4.0
error(CBS+MCB)/error(CBS)	$3.6 \cdot 10^{-5}$	1.3
error(CB+MCB)/error(CB)	$9.8 \cdot 10^{-5}$	1.1

TABLE 5. Comparison of zero finders applied to polynomials of degree 20 with zeros uniformly distributed in the square (7) with no pair of zeros closer than 0.2.

Method	Average Error	Smallest Error
SHB+MCB	$2.5 \cdot 10^{-5}$	72%
SHB	$4.4 \cdot 10^{-5}$	28%
CBS+MCB	$2.5 \cdot 10^{-5}$	73%
CBS	$1.4 \cdot 10^{-3}$	27%
CB+MCB	$1.2 \cdot 10^{-4}$	89%
CB	$3.0 \cdot 10^{-4}$	11%

TABLE 6. Comparison of zero finders applied to polynomials of degree 30 with zeros uniformly distributed in the unit disk (8) with no pair of zeros closer than 0.1.

Method	Average Error	Smallest Error
SHB+MCB	$4.5 \cdot 10^{-4}$	89%
SHB	$6.0 \cdot 10^{-3}$	11%
CBS+MCB	$4.4 \cdot 10^{-4}$	98%
CBS	$2.2 \cdot 10^{-2}$	2%
CB+MCB	$1.1 \cdot 10^{-3}$	100%
CB	$5.5 \cdot 10^{-2}$	0%

between zeros allows the eigenvalue problems to be more ill-conditioned. This has the effect that the average error in the computed zeros is larger than for Example 3.1. Table 3 shows the average error achieved by the different zero finders when applied to 100 polynomials and Table 4 displays error quotients. These tables are analogous to Tables 1 and 2, respectively. The CB+MCB method is seen to give the smallest average errors.  $\square$

Example 3.3. The polynomials generated in this example are of degree 20 with real or complex conjugate zeros uniformly distributed in the square (7). The distance between each pair of zeros is at least 0.2, as in Example 3.1. Table 5 shows the average errors achieved by the different methods. For all methods the average errors are smaller than in Table 1. The smallest average errors are achieved by the SHB+MCB and CBS+MCB methods.  $\square$

**Example 3.4.** The polynomials in this example are of degree 30 with real or complex conjugate zeros distributed uniformly in the unit disk (8). Similarly as in Example 3.2, the distance between each pair of zeros is at least 0.1. Table 6 shows the average errors achieved by the different methods. The smallest average errors are obtained by the SHB+MCB and CBS+MCB methods.  $\square$

#### 4. Conclusion

Numerous numerical experiments, some of which have been presented in Section 3, indicate that the CB, CBS and SHB methods for computing zeros of polynomials can be enhanced by the MCB method. For many polynomials, the highest accuracy is achieved by combining the SHB or CBS methods with the MCB method.

#### References

- [1] G. S. Ammar, D. Calvetti, W. B. Gragg, and L. Reichel, Polynomial zerofinders based on Szegő polynomials, *J. Comput. Appl. Math.*, 127 (2001), pp. 1–16.
- [2] A. Edelman and H. Murakami, Polynomial roots from companion matrix eigenvalues, *Math. Comp.*, 64 (1995), pp. 763–776.
- [3] S. Goedecker, Remark on algorithms to find roots of polynomials, *SIAM J. Sci. Comput.*, 15 (1994), pp. 1059–1063.
- [4] U. Grenander and G. Szegő, *Toeplitz Forms and Applications*, Chelsea, New York, 1984.
- [5] P. Henrici, *Applied and Computational Complex Analysis*, vol. 1, Wiley, New York, 1974.
- [6] M. A. Jenkins and J. F. Traub, A three-stage variable shift iteration for polynomial zeros and its relation to generalized Ritz iteration, *Numer. Math.*, 14 (1970), pp. 252–263.
- [7] F. Leja, Sur certaines suites liées aux ensembles plan et leur application à la représentation conforme, *Ann. Polon. Math.*, 4 (1957), pp. 8–13.
- [8] Matlab, version 6.1, The MathWorks, Inc., Natick, MA, 2001.
- [9] J. M. McNamee, An updated supplementary bibliography on roots of polynomials. *J. Comput. Appl. Math.*, 110 (1999), pp. 305–306. This reference discusses the bibliography. The actual bibliography is available at the web site <http://www.elsevier.nl/locate/cam>.
- [10] C. Moler, Cleve's corner: roots - of polynomials, that is, *The MathWorks Newsletter*, v. 5, n. 1 (1991), pp. 8–9.
- [11] B. Smith, J. Boyle, Y. Ikebe, V. Klema, and C. Moler, *Matrix Eigensystem Routines: EISPACK Guide*, Springer, Berlin, 2nd ed., 1976.
- [12] J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis*, 2nd ed., Springer, New York, 1993.

DEPARTMENT OF MATHEMATICS, CASE WESTERN RESERVE UNIVERSITY, CLEVELAND, OH 44106, USA

*E-mail address:* `dxc57@po.cwru.edu`

DEPARTMENT OF MATHEMATICAL SCIENCES, KENT STATE UNIVERSITY, KENT, OH 44242, USA

*E-mail address:* `reichel@mcs.kent.edu`

DIPARTIMENTO DI MATEMATICA, UNIVERSITÀ DI BOLOGNA, PIAZZA P.TA S. DONATO 5, BOLOGNA, ITALY

*E-mail address:* `sgallari@dm.unibo.it`

# A Fast Direct Method For Solving The Two-dimensional Helmholtz Equation, With Robbins Boundary Conditions

Jef Hendrickx, Raf Vandebril, and Marc Van Barel

**ABSTRACT.** We present a fast direct method for solving the two-dimensional Helmholtz equation:

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} + \lambda \phi = f(x, y),$$

on a rectangular grid  $[0, a_1] \times [0, a_2]$  with Robbins boundary conditions

$$\begin{cases} \left( \frac{\partial \phi}{\partial x} - p_0 \phi \right) (0, y) &= \alpha_0(y), \\ \left( \frac{\partial \phi}{\partial x} - p_1 \phi \right) (a_1, y) &= \alpha_1(y), \\ \left( \frac{\partial \phi}{\partial y} - q_0 \phi \right) (x, 0) &= \beta_0(x), \\ \left( \frac{\partial \phi}{\partial y} - q_1 \phi \right) (x, a_2) &= \beta_1(x), \end{cases}$$

where  $\lambda, p_0, p_1, q_0$  and  $q_1$  are constants.

Because we can solve the Helmholtz equation with Neumann boundary conditions in a fast way using the discrete cosine transform, we can split the problem above into two smaller problems. One of these problems can be solved using the same techniques as in the Neumann-boundary case. The second, and the hardest problem of the two, can be solved using low displacement rank techniques.

When dividing  $[0, a_1]$  into  $n_1$  and  $[0, a_2]$  into  $n_2$  equal parts, the total complexity of the overall algorithm is  $10n_1n_2 \log n_2 + O(n_1^2 + n_1n_2)$ , which gives us a fast algorithm.

## 1. Introduction

Several methods have been designed to solve the Poisson equation in a fast way. The Poisson equation is a special case of the Helmholtz equation when  $\lambda$  is equal to 0. When the Poisson equation is discretized using the classical five-point formula on a rectangular domain with Dirichlet boundary conditions on two opposite sides, Hockney [19, 20, 21] showed that the corresponding linear system can be solved using two types of direct methods: on the one hand the matrix decomposition method or Fourier analysis and on the other

---

1991 *Mathematics Subject Classification.* Primary 35J05; Secondary 65F05.

*Key words and phrases.* Helmholtz, Robbins boundary conditions, displacement rank, Neumann boundary conditions.

This research was partially supported by the Fund for Scientific Research–Flanders (FWO–V), project “SMA: Structured Matrices and their Applications” grant #G.0078.01, by the K.U.Leuven (Bijzonder Onderzoeksfonds), project “SLAP: Structured Linear Algebra Package,” grant #OT/00/16, by the Belgian Programme on Interuniversity Poles of Attraction, initiated by the Belgian State, Prime Minister’s Office for Science, Technology and Culture. The scientific responsibility rests with the authors.

hand the cyclic reduction method. The reader can find an excellent exposition in [41] by Swarztrauber who made several important contributions in this field. For an overview of the direct methods up to 1970 we refer the interested reader to the article of Dorr [12]. An even more detailed overview is given by Swarztrauber in [37] and by Buzbee, Golub and Concus [5]. The last article also describes the adaptations made by Buneman [4] to make the cyclic reduction method more stable.

Initially these methods were limited to a rectangular domain but now they are generalized to irregular domains [6, 7], the disc [36], the surface of a sphere [38]. The methods were also extended to separable [39] and non-separable [11] elliptic differential equations. Sweet [42] adapted the method of cyclic reduction to linear systems of arbitrary order instead of a power of 2.

Up to now, to our knowledge, no fast direct method was developed to solve the Helmholtz equation, with Robbins boundary conditions on all four sides. In this article we will design such a method and show the speed and accuracy of this method by some numerical experiments. However, the algorithm as presented here, is not competitive compared to the existing multigrid techniques for solving discretized PDE's. These techniques are faster, more precisely, they require  $O(n_1 n_2)$  ops (arithmetic operations) while our algorithm requires  $O(n_1 n_2 \log n_2)$  ops. However the computational efficiency is not the first aim of our algorithm. We want to show that structured matrix techniques can be used in a nice and natural way to solve these particular PDE problems. We hope that this approach eventually will lead to direct methods competitive to the multigrid ones. An overview of multigrid methods can be found in [15, 2, 44, 3]. We refer the reader who is interested in the combination of multigrid and structured matrix techniques to [34, 8, 9]. For a fast iterative solver based on FFT techniques, we refer to the work of Pickering and Harley [31, 32, 33]. For examples of applications the reader can have a look at [29].

The article is divided in three large parts, the first part deals with the Helmholtz equation, with Neumann boundary conditions. The solution of this problem is of main interest in solving the Helmholtz equation with Robbins boundary conditions, which we will take care of in the second section. The final section is dedicated to numerical experiments: we deduce some speed and accuracy results.

## 2. The Neumann boundary problem

In this section we will briefly review a fast method to solve the Helmholtz equation on a rectangle with Neumann boundary conditions. This problem can easily be solved using a fast transformation. As it will be shown in the next section, this is an important property in solving the Robbins boundary problem, in which we can twice use the techniques we are about to describe. In this point of view as mentioned in the abstract, the Robbins boundary problem can be split into two cheaper problems of which one can be solved using the Neumann boundary problem. First we will describe the solution to this problem, which is very similar to a method in the book of Pickering [30].

We take the Helmholtz equation,

$$(1) \quad \frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} + \lambda \phi = f(x, y),$$

on a rectangle  $[0, a_1] \times [0, a_2]$  with Neumann boundary conditions on two parallel sides, e.g. on the sides  $y = 0$  and  $y = a_2$ .

$$(2) \quad \frac{\partial \phi}{\partial y}(x, 0) = \beta_0(x), \quad \frac{\partial \phi}{\partial y}(x, a_2) = \beta_1(x), \quad 0 \leq x \leq a_1.$$

The boundary conditions on the two other sides  $x = 0$  and  $x = a_1$  are of Robbins type (with  $0 \leq y \leq a_2$ ),

$$(3) \quad \left( \frac{\partial \phi}{\partial x} - p_0 \phi \right)(0, y) = \alpha_0(y), \quad \left( \frac{\partial \phi}{\partial x} - p_1 \phi \right)(a_1, y) = \alpha_1(y).$$

We divide  $[0, a_1]$  (resp.  $[0, a_2]$ ) into  $n_1$  (resp.  $n_2$ ) equal parts. When using the classical five points difference scheme for  $i = 0, 1, \dots, n_1$  and  $j = 0, 1, \dots, n_2$ , we get

$$(4) \quad u_{i,j} - \theta_1(u_{i+1,j} + u_{i-1,j}) - \theta_2(u_{i,j+1} + u_{i,j-1}) = -\delta f_{i,j}$$

with

$$(5) \quad h_1 = \frac{a_1}{n_1}, \quad h_2 = \frac{a_2}{n_2}, \quad \delta = \left( \frac{2}{h_1^2} + \frac{2}{h_2^2} - \lambda \right)^{-1}, \quad \theta_1 = \frac{\delta}{h_1^2}, \quad \theta_2 = \frac{\delta}{h_2^2}$$

where  $u_{i,j}$  is the approximated value of  $\phi(x, y)$  in the grid point  $(x_i, y_j)$ . In the same way we define  $f_{i,j}$  as the value of  $f(x, y)$  in the grid point  $(x_i, y_j)$ . Approximating the boundary conditions with central differences we can substitute  $u_{i,-1}$  and  $u_{i,n_2+1}$  by

$$(6) \quad u_{i,-1} = u_{i,1} - 2h_2\beta_{0,i}, \quad u_{i,n_2+1} = u_{i,n_2-1} + 2h_2\beta_{1,i},$$

with  $\beta_{k,i} = \beta_k(x_i)$ . This gives us a system of equations  $M_{NN}\mathbf{u} = \mathbf{b}$  with unknowns  $u_{i,j}$  for  $i = 0, 1, \dots, n_1$  and  $j = 0, 1, \dots, n_2$  where

$$(7) \quad \mathbf{u} = \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{n_2} \end{bmatrix}, \quad \mathbf{u}_j = \begin{bmatrix} u_{0,j} \\ u_{1,j} \\ \vdots \\ u_{n_1,j} \end{bmatrix}.$$

and  $M_{NN}$  is the following block matrix:

$$(8) \quad M_{NN} = \begin{bmatrix} A & 2T & & \\ T & A & T & \\ & \ddots & \ddots & \ddots & \\ & & T & A & T \\ & & & 2T & A \end{bmatrix},$$

where  $A$  and  $T$  are of the form:  $T = -\theta_2 I$  and

$$(9) \quad A = \begin{bmatrix} 1 + 2h_1 p_0 \theta_1 & -2\theta_1 & & \\ -\theta_1 & 1 & -\theta_1 & \\ & \ddots & \ddots & \ddots & \\ & & -\theta_1 & 1 & -\theta_1 \\ & & & -2\theta_1 & 1 - 2h_1 p_1 \theta_1 \end{bmatrix},$$

with  $\theta_1$  and  $\theta_2$  as in (5).

It is a well-known fact that a matrix like

$$B = \begin{bmatrix} a & 2t & & \\ t & a & t & \\ & \ddots & \ddots & \ddots & \\ & & t & a & t \\ & & & 2t & a \end{bmatrix}$$

of size  $n + 1$  with  $a$  and  $t$  scalars, can be diagonalized in the following sense

$$(10) \quad B = C \Lambda C$$

with  $C = (c_{i,j})$  and  $\Lambda = \text{diag}\{\lambda_0, \dots, \lambda_n\}$ , where

$$c_{i,j} = \begin{cases} \sqrt{2/n} 1/2, & \text{for } j = 0, \quad i = 0, \dots, n, \\ \sqrt{2/n} \cos\left(\frac{ij\pi}{n}\right) & \text{for } j = 1, \dots, n-1, \quad i = 0, \dots, n, \\ \sqrt{2/n}(-1)^i 1/2, & \text{for } j = n, \quad i = 0, \dots, n, \end{cases}$$

and

$$\lambda_i = a + 2t \cos \frac{i\pi}{n}, \quad i = 0, \dots, n.$$

To prove this, write the matrix  $B$  as  $aI + tX_{00}$  with  $X_{00}$  as in (18), and then use the formula (27). A more compact form for the matrix  $C$  is

$$C = \sqrt{\frac{2}{n}} \left[ \varepsilon_j \cos \frac{ij\pi}{n} \right]_{i,j=0}^n,$$

with

$$(11) \quad \varepsilon_k = \begin{cases} 1/2 & \text{for } k = 0, n \\ 1 & \text{for } k = 1, 2, \dots, n-1 \end{cases}$$

(When we explicitly want to denote the size of the matrix  $C$  we denote it as  $C_{n+1}$ .) The matrix  $C$  is the matrix of one of the versions of the discrete cosine transform (*DCT*), [18, 1, 10, 35, 43]. This matrix is nor symmetric, nor persymmetric but satisfies  $(C)^{-1} = C$ .

Before we can continue we need some new notations. The symbol  $\otimes$  denotes the *Kronecker product* of matrices [14]. For a  $(p \times q)$ -matrix  $A = [a_{i,j}]$  and a  $(r \times s)$ -matrix  $B = [b_{i,j}]$  the Kronecker product  $C = B \otimes A$  is a  $(pr \times qs)$ -matrix defined as

$$C = \begin{bmatrix} b_{1,1}A & \cdots & b_{1,s}A \\ \vdots & & \vdots \\ b_{r,1}A & \cdots & b_{r,s}A \end{bmatrix}.$$

It is obvious that the matrix  $C$  has a block structure with  $r$  block rows and  $s$  block columns. We denote the  $(i_2, j_2)$ th block submatrix of  $C$  with  $C_{i_2, j_2}$ . The  $(i_1, j_1)$ th element of  $C_{i_2, j_2}$  is denoted as  $C_{i_1, j_1; i_2, j_2}$  and equals  $a_{i_1, j_1} b_{i_2, j_2}$ .

In a similar way as in the scalar case (10), it can be shown that the matrix  $M_{NN}$  can be factorised in the following way,

$$(12) \quad M_{NN} = (C \otimes I) \Lambda (C \otimes I),$$

with  $\Lambda$  a block diagonal matrix with blocks

$$(13) \quad \Lambda_j = A + 2T \cos \frac{j\pi}{n_2}, \quad j = 0, \dots, n_2.$$

We now present the algorithm for solving the Helmholtz equation with the Neumann boundary conditions. We use an operator called *Vec* which makes a long vector from a matrix by placing all the columns of the matrix one below the other.

**ALGORITHM 1.** *Solving  $M_{NN}\mathbf{u} = \mathbf{b}$  where  $M_{NN}$  is the block tridiagonal matrix of size  $(n_1 + 1) \times (n_2 + 1)$ , given by (8), with  $A$  and  $T$  of size  $n_1 + 1$  as in (9). Let  $\mathbf{b} = \text{vec } V$  with  $V \in \mathbb{R}^{(n_1+1) \times (n_2+1)}$  (and the same for  $\mathbf{z} = \text{vec } Z$ ,  $\mathbf{y} = \text{vec } Y$ ,  $\mathbf{u} = \text{vec } U$ ).*

- (1)  $Z = VC_{n_2+1}$
- (2) for  $j = 0, \dots, n_2$ :

solve the systems of equations  $\left(A + 2T \cos \frac{j\pi}{n_2}\right) \mathbf{y}_j = \mathbf{z}_j$  for  $\mathbf{y}_j$

- (3)  $U = YC_{n_2+1}$

The transformations in the first and the final step are cosine transformations on the rows of the matrix  $V$  (resp.  $Y$ ). Such a transformation of length  $n$  can be done using the Fast Fourier Transform (FFT) in  $2.5n \log n$  ops [40, 28]). So we get a complexity of  $5n_1n_2 \log n_2 + 8n_1n_2$ , because  $A$  and  $T$  are tridiagonal (under the assumption that  $n_2$  is a power of 2, or has at least small prime factors).

### 3. The Robbins boundary problem

We are now ready to deduce a fast direct method for solving the Helmholtz equation.

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} + \lambda \phi = f(x, y),$$

with Robbins boundary conditions, on all four sides.

$$(14) \quad \begin{cases} \left( \frac{\partial \phi}{\partial x} - p_0 \phi \right)(0, y) &= \alpha_0(y), \\ \left( \frac{\partial \phi}{\partial x} - p_1 \phi \right)(a_1, y) &= \alpha_1(y), \\ \left( \frac{\partial \phi}{\partial y} - q_0 \phi \right)(x, 0) &= \beta_0(x), \\ \left( \frac{\partial \phi}{\partial y} - q_1 \phi \right)(x, a_2) &= \beta_1(x). \end{cases}$$

We place an  $n_1 \times n_2$  grid on the rectangle  $[0, a_1] \times [0, a_2]$  and discretize the Helmholtz equation. The Robbins boundary conditions are approximated using central differences,

$$(15) \quad \begin{aligned} \frac{1}{2h_1}(u_{1,j} - u_{-1,j}) - p_0 u_{0,j} &\approx \alpha_{0,j}, \\ \frac{1}{2h_1}(u_{n_1+1,j} - u_{n_1-1,j}) - p_1 u_{n_1,j} &\approx \alpha_{1,j}, \\ \frac{1}{2h_2}(u_{i,1} - u_{i,-1}) - q_0 u_{i,0} &\approx \beta_{0,i}, \\ \frac{1}{2h_2}(u_{i,n_2+1} - u_{i,n_2-1}) - q_1 u_{i,n_2} &\approx \beta_{1,i}, \end{aligned}$$

where  $h_1 = a_1/n_1$  and  $h_2 = a_2/n_2$ . When using the classical five points difference scheme for the Helmholtz equation for  $i = 0, 1, \dots, n_1$  and  $j = 0, 1, \dots, n_2$ , we get

$$(16) \quad u_{i,j} - \theta_1(u_{i+1,j} + u_{i-1,j}) - \theta_2(u_{i,j+1} + u_{i,j-1}) = -\delta f_{i,j},$$

with  $\delta, \theta_1$  and  $\theta_2$  as in (5).

Using the equations (15) to eliminate the unknown parameters  $u_{-1,j}, u_{n_1+1,j}, u_{i,-1}$  and  $u_{i,n_2+1}$ , we get the linear system  $M_{RR}\mathbf{u} = \mathbf{b}$ , where

$$(17) \quad M_{RR} = \begin{bmatrix} A & 2T & & \\ T & A & T & \\ & \ddots & \ddots & \ddots & \\ & & T & A & T \\ & & & 2T & A \end{bmatrix} + 2h_2\theta_2 \begin{bmatrix} q_0 I & & & \\ & 0 & & \\ & & \ddots & \\ & & & 0 \\ & & & -q_1 I \end{bmatrix},$$

where  $T$  and  $A$  are the same as in (9) and  $\mathbf{u}$  as in (7). We can write  $M_{RR}$  in the following form:

$$M_{RR} = I \otimes A + X_{00} \otimes T + \Gamma \otimes I,$$

with

$$(18) \quad X_{00} = \begin{bmatrix} 0 & 2 & & \\ 1 & 0 & 1 & \\ & \ddots & \ddots & \ddots \\ & & 1 & 0 & 1 \\ & & & 2 & 0 \end{bmatrix}, \quad \Gamma = \begin{bmatrix} \gamma_0 & & & \\ & 0 & & \\ & & \ddots & \\ & & & 0 \\ & & & \gamma_1 \end{bmatrix}.$$

where  $\gamma_0 = 2h_2\theta_2q_0$  and  $\gamma_1 = -2h_2\theta_2q_1$ . It is not hard to see that the problem above can be written as

$$(19) \quad M_{RR} = M_{NN} + \Gamma \otimes I, \quad \text{with} \quad M_{NN} = I \otimes A + X_{00} \otimes T$$

which is exactly the matrix of the Neumann problem as described in the previous section. We rearrange the equation  $M_{RR}\mathbf{u} = \mathbf{b}$  as

$$(20) \quad \mathbf{u} = M_{NN}^{-1}\mathbf{b} - M_{NN}^{-1} \begin{bmatrix} \gamma_0\mathbf{u}_0 \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \gamma_1\mathbf{u}_{n_2} \end{bmatrix}.$$

It is obvious from the equation above that if we can find the two vectors  $\mathbf{u}_0 = (u_{i,0})_{i=0}^{n_1}$  and  $\mathbf{u}_{n_2} = (u_{i,n_2})_{i=0}^{n_1}$ , we can solve the problem by successively solving two  $M_{NN}$  problems, for which the algorithm can be found in the previous section.

#### 4. Finding $\mathbf{u}_0$ and $\mathbf{u}_{n_2}$ using low displacement rank theory

**4.1. The concept of displacement rank.** First we will give a very short introduction to displacement theory. Take  $F, A$  and  $R$  as  $m$ th order matrices, satisfying the following equation:

$$\nabla_{\{F,A\}}(R) = FR - RA = GB^T,$$

where  $G$  and  $B$  are of dimensions  $m \times r$ .  $\nabla_{\{F,A\}}(\cdot)$  is called the displacement operator, the matrices  $G$  and  $B$  are called the generators of the matrix  $R$ . We try to make  $r$  as small as possible (by choosing the right  $F$  and  $A$ ) and  $r$  is called the displacement rank of  $R$ .

We have to remark that there are also other definitions for displacement operators [23, 24], and there are several connections between classes of structured matrices with respect to different displacement operators. For further details we refer the interested reader to [25].

**4.2. An expression for the first and the last block of the first block column of  $M_{NN}^{-1}$ .** We introduce some new notations:  $B_{NN} = M_{NN}^{-1}$  and  $\mathbf{z} = M_{NN}^{-1}\mathbf{b}$ . When partitioning the matrices  $B_{NN}$  and  $M_{RR}$  in subblocks, the first and the last equation of (20) in  $\mathbf{u}_0$  and  $\mathbf{u}_{n_2}$  become:

$$(21) \quad \begin{cases} (I + \gamma_0 B_{0,0})\mathbf{u}_0 + \gamma_1 B_{0,n_2}\mathbf{u}_{n_2} = \mathbf{z}_0 \\ \gamma_0 B_{n_2,0}\mathbf{u}_0 + (I + \gamma_1 B_{n_2,n_2})\mathbf{u}_{n_2} = \mathbf{z}_{n_2} \end{cases}$$

Because  $B_{n_2,n_2} = B_{0,0}$  and  $B_{n_2,0} = B_{0,n_2}$ , (this will become clear later on, look at formula (22)) we only have to determine the matrices  $B_0 = B_{0,0}$  and  $B_{n_2} = B_{n_2,0}$  to solve the problem.

We could find them immediately by using the following formula:

$$M_{NN} \begin{bmatrix} B_0 \\ \star \\ B_{n_2} \end{bmatrix} = \begin{bmatrix} I \\ 0 \\ 0 \end{bmatrix}$$

but this would give us a computational cost of  $O((n_1 n_2)^3)$  ops, which is too high.

However, because the matrices are highly structured, block tridiagonal, and almost block Toeplitz, we can expect a low displacement rank, so we can solve the system of equations with a much lower complexity.

In the sequel, we'll see that some of the matrices involved can be diagonalized or block diagonalized by a discrete cosine transform; for our purpose this is enough. But these matrices can also be seen as generalized  $S$ -matrices. We refer the interested reader to [22, 26, 27].

As mentioned before and shown in [22] the matrix  $M_{NN}$  can be factorized in the following sense:

$$(22) \quad M_{NN} = (C \otimes I) \Lambda (C \otimes I),$$

with  $\Lambda$  a block diagonal matrix with blocks:

$$(23) \quad \Lambda_k = A + 2T \cos \frac{k\pi}{n_2}, \quad k = 0, \dots, n_2.$$

Because  $(C \otimes I)^{(-1)} = (C \otimes I)$  it is easy to deduce the following formula:

$$(24) \quad n_2 B_0 = \frac{1}{2} \Lambda_0^{-1} + \frac{1}{2} \Lambda_{n_2}^{-1} + \sum_{k=1}^{n_2-1} \Lambda_k^{-1} = \sum_{k=0}^{n_2} \Lambda_k^{-1},$$

where the double accent in the summation means that we have to multiply the first and the last term by 1/2. In a similar way we can find

$$(25) \quad n_2 B_{n_2} = R_{n_2} = \sum_{k=0}^{n_2} (-1)^k \Lambda_k^{-1}.$$

**4.3. Low displacement rank representation of  $\Lambda_k$  and  $\Lambda_k^{-1}$ .** The matrices  $\Lambda_k$  are tridiagonal matrices and can be inverted easily. On the other hand these matrices can be written as

$$\Lambda_k = M_k + E, \quad \text{with } E = \text{diag}(\epsilon_0, 0, \dots, 0, \epsilon_1),$$

where  $\epsilon_0 = 2h_1 p_0 \theta_1$ ,  $\epsilon_1 = -2h_1 p_1 \theta_1$  and

$$M_k = (A - E) + 2T \cos \frac{k\pi}{n_2},$$

which can be diagonalized by using the discrete cosine transform.

To compute  $\Lambda_k^{(-1)}$  we will use the displacement operator  $\nabla_{\{X_{00}, X_{00}\}}(\cdot)$ ,

$$(26) \quad \nabla_{\{X_{00}, X_{00}\}}(R) = X_{00}R - RX_{00}.$$

Unfortunately we cannot reconstruct the complete matrix  $R$  from the image of  $R$  under this displacement operator so we will have to keep some extra information in order to determine the matrix  $R$  uniquely [13, 23, 24, 25, 16].

First we will start by constructing the generators of the matrices  $\Lambda_k^{-1}$ . Note that the matrix  $X_{00}$  can be diagonalized by  $C$  ( $C$  is the matrix of the *DCT*), which means that

$$(27) \quad X_{00} = CD(\mathbf{c})C,$$

with,

$$(28) \quad D(\mathbf{c}) = \text{diag}\{\mathbf{c}\} = 2 \text{diag}\{1, \cos \frac{\pi}{n_1}, \dots, \cos \frac{(n_1-1)\pi}{n_1}, -1\}.$$

Because  $X_{00}$  as well as  $M_k$  can be diagonalized by the *DCT*, they commute with respect to multiplication and we find

$$\begin{aligned} \nabla_{\{X_{00}, X_{00}\}}(\Lambda_k) &= (X_{00}M_k - M_kX_{00}) + (X_{00}E - EX_{00}) \\ &= X_{00}E - EX_{00} \\ &= \begin{bmatrix} 0 & -2\epsilon_0 & & \\ \epsilon_0 & 0 & 0 & \\ & \ddots & \ddots & \ddots \\ & & 0 & 0 & \epsilon_1 \\ & & & -2\epsilon_1 & 0 \end{bmatrix} \end{aligned}$$

It is easily seen that the displacement rank of  $\Lambda_k$  is 4, and we can easily construct a possible choice for the generators:

$$\nabla_{\{X_{00}, X_{00}\}}(\Lambda_k) = -\epsilon_0(2\mathbf{e}_0\mathbf{e}_1^T - \mathbf{e}_1\mathbf{e}_0^T) - \epsilon_1(2\mathbf{e}_{n_1}\mathbf{e}_{n_1-1}^T - \mathbf{e}_{n_1-1}\mathbf{e}_{n_1}^T).$$

where  $\mathbf{e}_j$  denotes the  $j$ th unit vector of length  $n_1 + 1$ . From the last formula it follows easily that  $\Lambda_k^{-1}$  also has displacement rank 4, and we immediately find some generators:

$$(29) \quad \begin{aligned} X_{00}\Lambda_k^{-1} - \Lambda_k^{-1}X_{00} &= -\Lambda_k^{-1}(X_{00}\Lambda_k - \Lambda_kX_{00})\Lambda_k^{-1} \\ &= \epsilon_0\Lambda_k^{-1}(2\mathbf{e}_0\mathbf{e}_1^T - \mathbf{e}_1\mathbf{e}_0^T)\Lambda_k^{-1} + \epsilon_1\Lambda_k^{-1}(2\mathbf{e}_{n_1}\mathbf{e}_{n_1-1}^T - \mathbf{e}_{n_1-1}\mathbf{e}_{n_1}^T)\Lambda_k^{-1}. \end{aligned}$$

However, it is possible to find more convenient generators, using the following lemma.

**LEMMA 1.** *Take  $R_0$  to be an arbitrary invertible matrix of size  $n_1 + 1$  where  $R_0$  has the following structure*

$$R_0 = \begin{bmatrix} \beta & 2\alpha & 0 & \cdots & 0 \\ \alpha & * & * & \cdots & * \\ 0 & * & * & \cdots & * \\ \vdots & * & * & \cdots & * \\ 0 & * & * & \cdots & * \end{bmatrix},$$

*with  $\alpha$  an arbitrary number different from 0, then we have that*

$$R_0^{-1}(2\mathbf{e}_0\mathbf{e}_1^T - \mathbf{e}_1\mathbf{e}_0^T)R_0^{-1} = \frac{1}{\alpha}(\mathbf{f}_0\mathbf{e}_0^T - \mathbf{e}_0\mathbf{g}_0^T), \text{ with } \mathbf{f}_0 = R_0^{-1}\mathbf{e}_0 \text{ and } \mathbf{g}_0 = R_0^{-T}\mathbf{e}_0.$$

**Proof :** From the definition of  $\mathbf{f}_0$  and  $\mathbf{g}_0$  we immediately find that

$$(30) \quad R_0^{-1}(2\mathbf{e}_0\mathbf{e}_1^T - \mathbf{e}_1\mathbf{e}_0^T)R_0^{-1} = 2\mathbf{f}_0(\mathbf{e}_1^T R_0^{-1}) - (R_0^{-1}\mathbf{e}_1)\mathbf{g}_0^T.$$

Because the first column of  $R_0$  has the following form

$$(31) \quad R_0\mathbf{e}_0 = \beta\mathbf{e}_0 + \alpha\mathbf{e}_1,$$

we get that

$$R_0^{-1}\mathbf{e}_1 = \frac{1}{\alpha}(\mathbf{e}_0 - \beta R_0^{-1}\mathbf{e}_0) = \frac{1}{\alpha}(\mathbf{e}_0 - \beta\mathbf{f}_0).$$

Similarly, we derive that

$$(32) \quad \mathbf{e}_1^T R_0^{-1} = \frac{1}{2\alpha}(\mathbf{e}_0^T - \beta\mathbf{g}_0^T).$$

Replacing  $R_0^{-1}\mathbf{e}_1$  by (32) and  $\mathbf{e}_1^T R_0^{-1}$  by (32) in equation (30), leads to the desired result. ■

In the same way we can deduce that for an invertible matrix  $R_{n_1}$  of the form

$$R_{n_1} = \begin{bmatrix} * & * & \cdots & * & 0 \\ \vdots & & & \vdots & \vdots \\ * & \cdots & * & * & 0 \\ * & \cdots & * & * & \alpha \\ 0 & \cdots & 0 & 2\alpha & * \end{bmatrix}$$

with  $\alpha$  an arbitrary number different from zero, the following equation holds

$$R_{n_1}^{-1}(2\mathbf{e}_{n_1}\mathbf{e}_{n_1}^T - \mathbf{e}_{n_1-1}\mathbf{e}_{n_1}^T)R_{n_1}^{-1} = \frac{1}{\alpha}(\mathbf{f}_{n_1}\mathbf{e}_{n_1}^T - \mathbf{e}_{n_1}\mathbf{g}_{n_1}^T),$$

with  $\mathbf{f}_{n_1} = R_{n_1}^{-1}\mathbf{e}_{n_1}$  and  $\mathbf{g}_{n_1} = R_{n_1}^{-T}\mathbf{e}_{n_1}$ .

When  $R = \Lambda_k$  we only have to calculate  $\mathbf{f}_0$  and  $\mathbf{f}_{n_1}$ . Indeed, take the diagonal matrix  $D = \text{diag}\{1, 2, \dots, 2, 1\}$ , then its easy to check that  $\Lambda_k^T D = D\Lambda_k$ ,  $D\mathbf{e}_0 = \mathbf{e}_0$  and  $D\mathbf{e}_{n_1} = \mathbf{e}_{n_1}$ . Taking  $\mathbf{f}_0 = \Lambda_k^{-1}\mathbf{e}_0$  and  $\mathbf{f}_{n_1} = \Lambda_k^{-1}\mathbf{e}_{n_1}$ , we get

$$\Lambda_k^T(D\mathbf{f}_0) = D\Lambda_k\mathbf{f}_0 = D\mathbf{e}_0 = \mathbf{e}_0,$$

$$\Lambda_k^T(D\mathbf{f}_{n_1}) = D\Lambda_k\mathbf{f}_{n_1} = D\mathbf{e}_{n_1} = \mathbf{e}_{n_1}.$$

Now we can calculate  $\mathbf{g}_0$  and  $\mathbf{g}_{n_1}$  using  $\mathbf{f}_0$  and  $\mathbf{f}_{n_1}$ :

$$\mathbf{g}_0 = D\mathbf{f}_0, \quad \mathbf{g}_{n_1} = D\mathbf{f}_{n_1}.$$

Finally we can rewrite (29) and get

$$(33) \quad \nabla_{\{X_{00}, X_{00}\}}(\Lambda_k^{-1}) = \frac{\varepsilon_0}{\theta_1} (\mathbf{x}_k\mathbf{e}_0^T - \mathbf{e}_0(D\mathbf{x}_k)^T) + \frac{\varepsilon_1}{\theta_1} (\mathbf{y}_k\mathbf{e}_{n_1}^T - \mathbf{e}_{n_1}(D\mathbf{y}_k)^T),$$

with

$$(34) \quad \mathbf{x}_k = \Lambda_k^{-1}\mathbf{e}_0 \quad \text{and} \quad \mathbf{y}_k = \Lambda_k^{-1}\mathbf{e}_{n_1}.$$

**4.4. A low displacement rank representation of  $B_0$  and  $B_{n_2}$ .** Using the equations (24) and (25) we find

$$(35) \quad X_{00}B_0 - B_0X_{00} = \mathbf{x}_+\mathbf{e}_0^T - \mathbf{e}_0(D\mathbf{x}_+)^T + \mathbf{y}_+\mathbf{e}_{n_1}^T - \mathbf{e}_{n_1}(D\mathbf{y}_+)^T$$

with

$$(36) \quad \mathbf{x}_+ = \left( \frac{-1}{n_2} \right) \left( \frac{\varepsilon_0}{\theta_1} \sum_{k=0}^{n_2} \mathbf{x}_k \right) \quad \text{and} \quad \mathbf{y}_+ = \left( \frac{-1}{n_2} \right) \left( \frac{\varepsilon_1}{\theta_1} \sum_{k=0}^{n_2} \mathbf{y}_k \right)$$

We see now that the matrix  $B_0$  has displacement rank 4, and we can find the generators using the formulas here above. In a completely similar way we can deduce the following relations for  $B_{n_2}$

$$(37) \quad \nabla_{\{X_{00}, X_{00}\}}(B_{n_2}) = \mathbf{x}_-\mathbf{e}_0^T - \mathbf{e}_0(D\mathbf{x}_-)^T + \mathbf{y}_-\mathbf{e}_{n_1}^T - \mathbf{e}_{n_1}(D\mathbf{y}_-)^T$$

with

$$\mathbf{x}_- = \left( \frac{-1}{n_2} \right) \left( \frac{\varepsilon_0}{\theta_1} \sum_{k=0}^{n_2} (-1)^k \mathbf{x}_k \right) \quad \text{and} \quad \mathbf{y}_- = \left( \frac{-1}{n_2} \right) \left( \frac{\varepsilon_1}{\theta_1} \sum_{k=0}^{n_2} (-1)^k \mathbf{y}_k \right)$$

Hence both  $B_0$  and  $B_{n_2}$  have low displacement rank, and we can calculate their generators.

**4.5. Solving a linear system having low  $\{X_{00}, X_{00}\}$  displacement rank.** Before solving the special case (21), we take a look at the more general problem: how can we solve a system of equations with a low  $\{X_{00}, X_{00}\}$  displacement rank? We take  $R$  to be an arbitrary square matrix of size  $n+1$ , for which

$$\nabla_{\{X_{00}, X_{00}\}}(R) = FG^T,$$

with  $F$  and  $G$   $(n+1) \times r$ -matrices and  $r \ll n$ . We already know that we can diagonalize  $X_{00}$  by using a cosine transformation.

Using the same techniques as in [13, 18] we can take another displacement operator, and transform  $R$  into another class of structured matrices. First of all we will give the definition of a generalized Cauchy matrix.

**DEFINITION 1.** Let  $c = (c_i)_1^n$  and  $d = (d_j)_1^n$  be fixed  $n$ -tuples of numbers and let  $A = [a_{ij}]_{i,j=1}^n$  be a given matrix. The Cauchy rank of a matrix is the rank  $r$  of

$$\nabla_{c,d}(A) = [(c_i - d_j)a_{ij}]_{i,j=1}^n.$$

When  $r$  is small with respect to the order of  $A$ , then  $A$  will be called a generalized Cauchy matrix. For a classical Cauchy matrix  $(c_i - d_j)a_{ij} = 1$  for all  $i$  and  $j$ , and has therefore, Cauchy rank 1.

When we take  $\hat{R} = CRC$ , then according to (26) and (27) we find that

$$(38) \quad \nabla_{\{D(\mathbf{c}), D(\mathbf{c})\}}(\hat{R}) = D(\mathbf{c})\hat{R} - \hat{R}D(\mathbf{c}) = \hat{F}\hat{G}^T$$

with  $\hat{F} = CF$  and  $\hat{G} = C^TG$ . Hence,  $\hat{R}$  is a generalized Cauchy matrix. Using the generators we can almost completely reconstruct the matrix  $\hat{R}$ . Only the diagonal elements are lost. The diagonal of  $\hat{R}$  can be constructed by first calculating  $\mathbf{u} = \hat{R}\mathbf{1} = CRC\mathbf{1}$  with  $\mathbf{1} = [1 \ \dots \ 1]^T$ . We can find the diagonal of  $\hat{R}$  using the following formula:

$$d_i = u_i - \sum_{j \neq i}^{0,n_1} \frac{\hat{\mathbf{f}}_i^T \hat{\mathbf{g}}_j}{c_i - c_j}, \quad i = 0, \dots, n_1.$$

We can now apply the LU-CAUCHY algorithm of Bojanczyk and Heinig [17] to construct the LU factorization of the matrix  $\hat{R}$  knowing the generators and its diagonal elements. We can also implement row pivoting in this algorithm [13].

**4.6. Computing  $\mathbf{u}_0$  and  $\mathbf{u}_{n_2}$ .** In this final paragraph, we will solve the equations (21). The matrix of the system of equations is

$$(39) \quad B = \begin{pmatrix} I + \gamma_0 B_0 & \gamma_1 B_{n_2} \\ \gamma_0 B_{n_2} & I + \gamma_1 B_0 \end{pmatrix}.$$

The four subblocks of the matrix all have a low  $\{X_{00}, X_{00}\}$ -displacement rank. This means that the matrix  $B$  will have a low displacement rank according to the following displacement operator  $\nabla_{\{X, X\}}$  with  $X$  the direct sum of  $X_{00}$  with itself:

$$X = X_{00} \oplus X_{00} = \begin{pmatrix} X_{00} & 0 \\ 0 & X_{00} \end{pmatrix}.$$

According to (27) we have

$$X_{00} \oplus X_{00} = (C \oplus C)(D(\mathbf{c}) \oplus D(\mathbf{c}))(C \oplus C).$$

Therefore, we can transform  $B$  to a generalized Cauchy-like matrix  $\hat{B}$ :

$$\hat{B} = (C \oplus C)B(C \oplus C),$$

using the displacement operator

$$(40) \quad \nabla_{\{D(\mathbf{c}) \oplus D(\mathbf{c}), D(\mathbf{c}) \oplus D(\mathbf{c})\}}(\cdot).$$

Note that

$$\hat{B} = \begin{pmatrix} I + \gamma_0 \hat{B}_0 & \gamma_1 \hat{B}_{n_2} \\ \gamma_0 \hat{B}_{n_2} & I + \gamma_1 \hat{B}_0 \end{pmatrix},$$

so we can calculate the generators of  $\hat{B}$  using these of  $\hat{B}_0 = CB_0C$  and  $\hat{B}_{n_2} = CB_{n_2}C$ . By multiplying (35) on both sides with  $C$ , using (27) and the fact that  $C^T D = DC$  we obtain the following result

$$(41) \quad \nabla_{\{D(\mathbf{c}), D(\mathbf{c})\}}(\hat{B}_0) = \hat{\mathbf{x}}_+(D\hat{\mathbf{e}}_0)^T - \hat{\mathbf{e}}_0(D\hat{\mathbf{x}}_+)^T + \hat{\mathbf{y}}_+(D\hat{\mathbf{e}}_{n_1})^T - \hat{\mathbf{e}}_{n_1}(D\hat{\mathbf{y}}_+)^T$$

and in the same way we get

$$(42) \quad \nabla_{\{D(\mathbf{c}), D(\mathbf{c})\}}(\hat{B}_{n_2}) = \hat{\mathbf{x}}_-(D\hat{\mathbf{e}}_0)^T - \hat{\mathbf{e}}_0(D\hat{\mathbf{x}}_-)^T + \hat{\mathbf{y}}_-(D\hat{\mathbf{e}}_{n_1})^T - \hat{\mathbf{e}}_{n_1}(D\hat{\mathbf{y}}_-)^T,$$

with  $\hat{\mathbf{x}}_{\pm} = C\mathbf{x}_{\pm}$ ,  $\hat{\mathbf{y}}_{\pm} = C\mathbf{y}_{\pm}$ ,  $\hat{\mathbf{e}}_i = C\mathbf{e}_i$ , for  $i = 0, \dots, n_1$ . We can now write the displacement of  $\hat{B}$  as

$$\nabla_{\{D(\mathbf{c}) \oplus D(\mathbf{c}), D(\mathbf{c}) \oplus D(\mathbf{c})\}}(\hat{B}) = \begin{pmatrix} \gamma_0 \nabla_{\{D(\mathbf{c}), D(\mathbf{c})\}}(\hat{B}_0) & \gamma_1 \nabla_{\{D(\mathbf{c}), D(\mathbf{c})\}}(\hat{B}_{n_2}) \\ \gamma_0 \nabla_{\{D(\mathbf{c}), D(\mathbf{c})\}}(\hat{B}_{n_2}) & \gamma_1 \nabla_{\{D(\mathbf{c}), D(\mathbf{c})\}}(\hat{B}_0) \end{pmatrix},$$

and we can calculate the generators

$$\begin{aligned} & \nabla_{\{D(\mathbf{c}) \oplus D(\mathbf{c}), D(\mathbf{c}) \oplus D(\mathbf{c})\}}(\hat{B}) \\ &= \gamma_0 \begin{pmatrix} \hat{\mathbf{x}}_+ \\ \hat{\mathbf{x}}_- \end{pmatrix} \begin{pmatrix} (D\hat{\mathbf{e}}_0)^T & \mathbf{0} \end{pmatrix} - \begin{pmatrix} \hat{\mathbf{e}}_0 \\ \mathbf{0} \end{pmatrix} \begin{pmatrix} \gamma_0(D\hat{\mathbf{x}}_+)^T & \gamma_1(D\hat{\mathbf{x}}_-)^T \end{pmatrix} \\ &+ \gamma_0 \begin{pmatrix} \hat{\mathbf{y}}_+ \\ \hat{\mathbf{y}}_- \end{pmatrix} \begin{pmatrix} (D\hat{\mathbf{e}}_{n_1})^T & \mathbf{0} \end{pmatrix} - \begin{pmatrix} \hat{\mathbf{e}}_{n_1} \\ \mathbf{0} \end{pmatrix} \begin{pmatrix} \gamma_0(D\hat{\mathbf{y}}_+)^T & \gamma_1(D\hat{\mathbf{y}}_-)^T \end{pmatrix} \\ &+ \gamma_1 \begin{pmatrix} \hat{\mathbf{x}}_- \\ \hat{\mathbf{x}}_+ \end{pmatrix} \begin{pmatrix} \mathbf{0} & (D\hat{\mathbf{e}}_0)^T \end{pmatrix} - \begin{pmatrix} \mathbf{0} \\ \hat{\mathbf{e}}_0 \end{pmatrix} \begin{pmatrix} \gamma_0(D\hat{\mathbf{x}}_-)^T & \gamma_1(D\hat{\mathbf{x}}_+)^T \end{pmatrix} \\ &+ \gamma_1 \begin{pmatrix} \hat{\mathbf{y}}_- \\ \hat{\mathbf{y}}_+ \end{pmatrix} \begin{pmatrix} \mathbf{0} & (D\hat{\mathbf{e}}_{n_1})^T \end{pmatrix} - \begin{pmatrix} \mathbf{0} \\ \hat{\mathbf{e}}_{n_1} \end{pmatrix} \begin{pmatrix} \gamma_0(D\hat{\mathbf{y}}_-)^T & \gamma_1(D\hat{\mathbf{y}}_+)^T \end{pmatrix} \end{aligned}$$

In order to reconstruct the matrix  $\hat{B}$  completely, besides the generators, we also need the diagonals of  $\hat{B}_0$  and  $\hat{B}_{n_2}$ . Note that we need to reconstruct the diagonals of the four subblocks of the matrix, to completely restore  $\hat{B}$ . Therefore, we have to adapt the LU-CAUCHY algorithm to this special situation.

To reconstruct the diagonal  $\mathbf{d}_+$  of  $\hat{B}_0$ , we first calculate  $\mathbf{u} = \hat{B}_0 \mathbf{1}$ , which we can rewrite using (24) as

$$\mathbf{u} = \left( \frac{1}{n_2} \right) \left( \sum_{k=0}^{n_2} \hat{\Lambda}_k^{-1} \mathbf{1} \right), \quad \text{with } \hat{\Lambda}_k^{-1} = C \Lambda_k^{-1} C.$$

Note that  $\mathbf{1} = \sqrt{2n_1} \hat{\mathbf{e}}_0$  and when we denote  $\hat{\mathbf{x}}_k = C\mathbf{x}_k$  we find using (34) that  $\hat{\mathbf{x}}_k = \hat{\Lambda}_k^{-1} \hat{\mathbf{e}}_0$ , such that

$$\mathbf{u} = \left( \frac{\sqrt{2n_1}}{n_2} \right) \sum_{k=0}^{n_2} \hat{\Lambda}_k^{-1} \hat{\mathbf{e}}_0 = \left( \frac{\sqrt{2n_1}}{n_2} \right) \sum_{k=0}^{n_2} \hat{\mathbf{x}}_k.$$

Because of (36) we get

$$\mathbf{u} = -\sqrt{2n_1} \frac{\theta_1}{\varepsilon_0} \hat{\mathbf{x}}_+.$$

In order to find the  $i$ th component of the diagonal  $\mathbf{d}_+$  we have to subtract from  $u_i$  the nondiagonal elements of the  $i$ th row of  $\hat{B}_0$  which we can find using the generators of  $\hat{B}_0$ . Finally we find for  $i = 0, \dots, n_1$

$$d_{+i} = -\sqrt{2n_1} \frac{\theta_1}{\varepsilon_0} \hat{x}_{+i} - \sqrt{\frac{2}{n_1}} \sum_{j \neq i}^{0, n_1} \frac{\hat{x}_{+i} - \hat{x}_{+j} + (-1)^j \hat{y}_{+i} - (-1)^i \hat{y}_{+j}}{c_i - c_j}.$$

In a completely analogous manner we can find the diagonal  $\mathbf{d}_-$  of  $\hat{B}_{n_2}$

$$d_{-i} = -\sqrt{2n_1} \frac{\theta_1}{\varepsilon_0} \hat{x}_{-i} - \sqrt{\frac{2}{n_1}} \sum_{j \neq i}^{0, n_1} \frac{\hat{x}_{-i} - \hat{x}_{-j} + (-1)^j \hat{y}_{-i} - (-1)^i \hat{y}_{-j}}{c_i - c_j}.$$

In the next section we give the algorithm summarizing all the previous steps.

## 5. The Algorithm

First of all we present the algorithm which calculates the LU factorization of the matrix  $\hat{B} = (C \oplus C)B(C \oplus C)$  with  $B$  given by (39). This LU-factorization can later on be used to solve (21).

**ALGORITHM 2.** *Calculating the LU factorization of the matrix  $B$  given by (39).*

- (1) Calculate the generators of the matrices  $\Lambda_k^{-1}$  (cf. formula (33)).  
For  $k = 0, \dots, n_2$ , calculate

$$\mathbf{x}_k = \Lambda_k^{-1} \mathbf{e}_0, \quad \mathbf{y}_k = \Lambda_k^{-1} \mathbf{e}_{n_1}.$$

- (2) Calculate the generators for  $B_0$  and  $B_{n_2}$  (cf. formula (35)).

Calculate

$$\begin{aligned} \mathbf{x}_+ &= \frac{(-1)}{n_2} \left( \frac{\varepsilon_0}{\theta_1} \sum_{k=0}^{n_2} \mathbf{x}_k \right), & \mathbf{x}_- &= \frac{(-1)}{n_2} \left( \frac{\varepsilon_0}{\theta_1} \sum_{k=0}^{n_2} (-1)^k \mathbf{x}_k \right), \\ \mathbf{y}_+ &= \frac{(-1)}{n_2} \left( \frac{\varepsilon_1}{\theta_1} \sum_{k=0}^{n_2} \mathbf{y}_k \right), & \mathbf{y}_- &= \frac{(-1)}{n_2} \left( \frac{\varepsilon_1}{\theta_1} \sum_{k=0}^{n_2} (-1)^k \mathbf{y}_k \right). \end{aligned}$$

- (3) Transform to the generators for  $\hat{B}_0$  and  $\hat{B}_{n_2}$  (cf. formula (41) and (42)).

Calculate

$$\begin{aligned} \hat{\mathbf{x}}_+ &= C \mathbf{x}_+, & \hat{\mathbf{x}}_- &= C \mathbf{x}_-, \\ \hat{\mathbf{y}}_+ &= C \mathbf{y}_+, & \hat{\mathbf{y}}_- &= C \mathbf{y}_-, \\ \hat{\mathbf{e}}_0 &= C \mathbf{e}_0, & \hat{\mathbf{e}}_{n_1} &= C \mathbf{e}_{n_1}. \end{aligned}$$

- (4) Calculate the diagonal  $\mathbf{d}$  of  $\hat{B}$  and the diagonals  $\mathbf{d}_U$  and  $\mathbf{d}_L$  of the  $(1,2)$ - and  $(2,1)$ -subblock of  $\hat{B}$  respectively.

Calculate for  $i = 1, \dots, n_1$ :

$$d_{+i} = -\sqrt{2n_1} \frac{\theta_1}{\varepsilon_0} \hat{x}_{+i} - \sqrt{\frac{2}{n_1}} \sum_{j \neq i}^{0, n_1} \frac{\hat{x}_{+i} - \hat{x}_{+j} + (-1)^j \hat{y}_{+i} - (-1)^i \hat{y}_{+j}}{c_i - c_j},$$

$$d_{-i} = -\sqrt{2n_1} \frac{\theta_1}{\varepsilon_0} \hat{x}_{-i} - \sqrt{\frac{2}{n_1}} \sum_{j \neq i}^{0, n_1} \frac{\hat{x}_{-i} - \hat{x}_{-j} + (-1)^j \hat{y}_{-i} - (-1)^i \hat{y}_{-j}}{c_i - c_j}.$$

$$\mathbf{d} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} + \begin{pmatrix} \gamma_0 \mathbf{d}_+ \\ \gamma_1 \mathbf{d}_+ \end{pmatrix}, \quad \mathbf{d}_L = \gamma_0 \mathbf{d}_-, \quad \mathbf{d}_U = \gamma_1 \mathbf{d}_-,$$

where  $c_i = c_{i+n_1+1} = 2 \cos \frac{i\pi}{n_1}$ , for  $i = 0, 1, \dots, n_1$ .

- (5) Initialise the generators  $\hat{F}$  and  $\hat{G}$  of  $\hat{B}$ .

Use the following notations

$$\begin{aligned} \nabla_{\{D(\mathbf{c}) \oplus D(\mathbf{c}), D(\mathbf{c}) \oplus D(\mathbf{c})\}}(\hat{B}) &= \hat{F} \hat{G}^T \\ &= \begin{pmatrix} \phi_0^T \\ \phi_1^T \\ \vdots \\ \phi_{2n_1+1}^T \end{pmatrix} \begin{pmatrix} \psi_0 & \psi_1 & \dots & \psi_{2n_1+1} \end{pmatrix} \end{aligned}$$

Calculate  $\hat{F}$  and  $\hat{G}$  as

$$\hat{F} = \begin{bmatrix} \gamma_0 \hat{\mathbf{x}}_+ & \gamma_0 \hat{\mathbf{y}}_+ & -\hat{\mathbf{e}}_0 & -\hat{\mathbf{e}}_{n_1} & \gamma_1 \hat{\mathbf{x}}_- & \gamma_1 \hat{\mathbf{y}}_- & \mathbf{0} & \mathbf{0} \\ \gamma_0 \hat{\mathbf{x}}_- & \gamma_0 \hat{\mathbf{y}}_- & \mathbf{0} & \mathbf{0} & \gamma_1 \hat{\mathbf{x}}_+ & \gamma_1 \hat{\mathbf{y}}_+ & -\hat{\mathbf{e}}_0 & -\hat{\mathbf{e}}_{n_1} \end{bmatrix}$$

and

$$\hat{G} = \begin{bmatrix} D\hat{\mathbf{e}}_0 & D\hat{\mathbf{e}}_{n_1} & \gamma_0 D\hat{\mathbf{x}}_+ & \gamma_0 D\hat{\mathbf{y}}_+ & \mathbf{0} & \mathbf{0} & \gamma_0 D\hat{\mathbf{x}}_- & \gamma_0 D\hat{\mathbf{y}}_- \\ \mathbf{0} & \mathbf{0} & \gamma_1 D\hat{\mathbf{x}}_- & \gamma_1 D\hat{\mathbf{y}}_- & D\hat{\mathbf{e}}_0 & D\hat{\mathbf{e}}_{n_1} & \gamma_1 D\hat{\mathbf{x}}_+ & \gamma_1 D\hat{\mathbf{y}}_+ \end{bmatrix}$$

- (6) The LU-CAUCHY algorithm adapted to this situation (cf. [17]).

For  $k = 0, 1, \dots, 2n_1 + 1$ , calculate

- (a) Calculate column  $k$  of  $L$  and row  $k$  of  $U$  from the generators.

$$l_{kk} = d_k$$

For  $j = k+1, k+2, \dots, 2n_1+1$ ,

$$(i) \quad l_{jk} = \begin{cases} d_{Lk} & \text{if } j = n_1 + 1 + k \\ \frac{\phi_j^T \psi_k}{c_j - c_k} & \text{else} \end{cases}$$

$$(ii) \quad u_{kj} = \begin{cases} d_{Uk} & \text{if } j = n_1 + 1 + k \\ \frac{\phi_k^T \psi_j}{c_k - c_j} & \text{else} \end{cases}$$

$$u_{kk} = l_{kk} \text{ and } l_{kk} = 1.$$

For  $j = k+1, k+2, \dots, 2n_1+1$ , calculate  $l_{jk} = \frac{l_{jk}}{u_{kk}}$

- (b) Calculate the Schurcomplement.

For  $j = k+1, k+2, \dots, 2n_1+1$ , calculate

$$(i) \quad \phi_j = \phi_j - l_{jk} \phi_k$$

$$(ii) \quad t = \frac{u_{kj}}{u_{kk}}$$

$$(iii) \quad \psi_j = \psi_j - t \psi_k$$

$$(iv) \quad d_j = d_j - l_{jk} u_{kj}$$

$$(v) \quad \text{If } k \leq n_1,$$

$$d_{Lj} = d_{Lj} - l_{n_1+1+j, k} u_{kj}$$

$$d_{Uj} = d_{Uj} - l_{jk} u_{k, n_1+1+j}.$$

In the same way as in [13] we can apply row pivoting after step 6(a)i. This will improve the numerical stability of the algorithm. The complexity of the algorithm presented here is  $O(n_1 n_2) + O(n_1^2)$ . Indeed, step 1 requires solving  $n_2 + 1$  tridiagonal systems of equations of size  $n_1 + 1$ , in step 2 and 4 we have to add  $n_2 + 1$  vectors of length  $n_1 + 1$  and in the final

step for every  $k$  (from 0 to  $n_1$ )  $72(2n_1 - k)$  ops have to be performed. The complexity of the other steps is of a lower magnitude.

When we want to solve (21) we use the fact that

$$B = (C \oplus C) \hat{B} (C \ominus C)$$

So we get the following algorithm for the solution of the Helmholtz equation with Robbins boundary conditions.

**ALGORITHM 3** (Robbins problem). *This algorithm solves the system of equations*

$$M_{RR}\mathbf{u} = \mathbf{b}$$

where  $M_{RR}$  is the matrix (17) coming from a finite difference approximation of the Helmholtz equation on a rectangular grid with Robbins boundary conditions.

- (1) Find the LU-factorization of  $\hat{B}$  using the previous algorithm (Algorithm 2).
- (2) Calculate  $\mathbf{z} = M_{NN}^{-1}\mathbf{b}$  with  $M_{NN}$  from (19) using a fast direct method.
- (3) Calculate  $\mathbf{u}_0$  and  $\mathbf{u}_{n_2}$ .
  - (a) Calculate  $\hat{\mathbf{z}}_0 = C\mathbf{z}_0$  and  $\hat{\mathbf{z}}_{n_2} = C\mathbf{z}_{n_2}$  with a fast cosine transform.
  - (b) Solve the system of equations

$$\hat{B} \begin{bmatrix} \hat{\mathbf{u}}_0 \\ \hat{\mathbf{u}}_{n_2} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{z}}_0 \\ \hat{\mathbf{z}}_{n_2} \end{bmatrix}$$

by using the LU-factorization of  $\hat{B}$ .

- (c) Calculate  $\mathbf{u}_0 = C\hat{\mathbf{u}}_0$  and  $\mathbf{u}_{n_2} = C\hat{\mathbf{u}}_{n_2}$  with a fast cosine transform.
- (4) Calculate

$$\mathbf{y} = M_{NN}^{-1} \begin{bmatrix} \gamma_0 \mathbf{u}_0 \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \gamma_{n_2} \mathbf{u}_{n_2} \end{bmatrix}$$

analogous as in step 2.

- (5) Calculate  $\mathbf{u} = \mathbf{y} - \mathbf{z}$

Because we had to solve two systems of equations with  $M_{NN}$  as coefficient matrix, the computational complexity of the algorithm will be  $10n_1 n_2 \log n_2 + O(n_1^2 + n_1 n_2)$  ops. The iterative technique of Pickering and Harley [32], has complexity:  $A(n_1)n_2^2 + O(n_2^2 \log_2(n_2))$ , where  $A$  is proportional to the number of iterations.

## 6. Numerical Examples

**6.1. Timings.** We implemented the algorithm (with pivoting) in Fortran90, using the NAG95-Fortran compiler on a linux platform. The tests were done on a Pentium III 860 Mhz processor, with 512 MByte of memory. In order to test its complexity, we generated a fixed example and changed the variables  $n_1$  and  $n_2$ . By using the least squares approximation we obtained the following complexity:

$$0.6n_1 n_2 \log(n_2) + 3.6n_1^2 - 3.6n_1 n_2$$

in ( $10^{-6}$  sec). It can be seen in figure 1 that this is a good approximation: the surface is generated using the formula above, and the dots are the measured timings. Making  $n_1$  larger, slows down the algorithm more than enlarging the factor  $n_2$ . However this is no problem, because, when  $n_1$  is larger then  $n_2$ , we can just swap the roles of the variables  $x$  and  $y$  in (1). Then also  $n_1$  and  $n_2$  are interchanged.

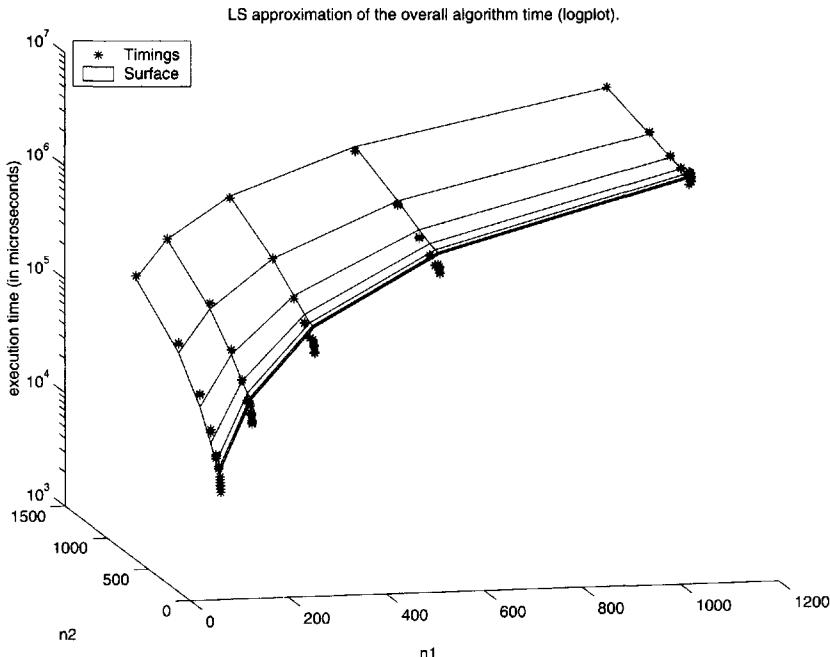


FIGURE 1. Timing results for the overall algorithm.

**6.2. Stability.** First of all, we changed the parameter  $\lambda$ , to make the system of equations very ill-conditioned, we did this for parameter values  $n_1 = 16$  and  $n_2 = 16$ . We also took  $p_0, p_1, q_0, q_1$  to be different from 0, because otherwise we would have the Neumann problem. Note that if we fill in for  $\lambda$  the eigenvalues of  $M_{RR}$  (with  $\lambda=0$ ), then the corresponding  $M_{RR}$  matrix becomes singular. We approximated the largest eigenvalue in modulus using the power method, and then changed it slightly, to obtain some different condition numbers for the matrix  $M_{RR}$ . The largest eigenvalue obtained by the power method was  $\lambda = 1.06175777127e + 02$ . In the following table:  $\Delta\lambda$  and the different condition numbers are shown:

$\Delta\lambda$	Condition( $M_{RR}$ )	Condition( $M_{NN}$ )	Condition( $B$ )
6.2500000e-02	1.9730530e+03	1.0973156e+02	9.8528442e+01
7.7160499e-04	1.5987623e+05	1.1662279e+02	7.8476879e+03
1.1972999e-05	1.0303353e+07	1.1671302e+02	5.0564809e+05
1.1800000e-07	1.0494080e+09	1.1671443e+02	5.1500656e+07
9.9998942e-10	1.0242767e+11	1.1671445e+02	5.0267124e+09
0	1.0270375e+13	1.1671445e+02	5.0399766e+11

It can be seen, that the bad conditioning of  $M_{RR}$  is completely incorporated in the matrix  $B$ . In the following table, it can be seen that the loss of significant digits can be explained completely by the ill-conditioning of the matrix. When using the notation  $M_{RR}\mathbf{u} = \mathbf{b}$ , where  $\mathbf{u}$  denotes the exact solution and  $\mathbf{x}$  denotes the solution calculated by the algorithm, then the relative residual norm is calculated as follows:  $\|M_{RR}\mathbf{x} - \mathbf{b}\|_2 / \|\mathbf{b}\|_2$ .

Relative residual norm	$\ x\ _2$	$\ M_{RR}\ _2$
0.269875541131E-15	0.137537825554E+02	2.124455918105E+00
0.151802077218E-13	0.750048621793E+03	2.125753734878E+00
0.775739556738E-12	0.483388831755E+05	2.125769724492E+00
0.787803224555E-10	0.492337955471E+07	2.125769974033E+00
0.745524749715E-08	0.480537415275E+09	2.125769976496E+00
0.938779346662E-06	0.482384800059E+11	2.125769976517E+00

In the second experiment we first took fixed parameters, for the system  $M_{RR}$ , except for  $\lambda$  and the right-hand side  $f_{i,j}$ , the  $\lambda$ 's here are the same as those above, to have an ill-conditioned system. For every  $\lambda$  we computed the right-hand side using a fixed solution  $\mathbf{u}_e$  (with  $\|\mathbf{u}_e\|_2 \approx 500$ ). On these problems we ran the algorithm, and calculated the relative residual norm and the relative error norm (which is calculated as follows:  $\|\mathbf{x} - \mathbf{u}_e\|_2 / \|\mathbf{u}_e\|_2$ ), as shown in the next table:

Relative residual norm	Relative error norm
0.247972802994E-15	0.273041336350E-13
0.302336805921E-15	0.107901396342E-11
0.236340232958E-15	0.163183467236E-09
0.308345718370E-15	0.269762204052E-08
0.251476127414E-15	0.530808215848E-07
0.269757268817E-15	0.707838423187E-04

In the last test we did, we tried some random examples. We generated random examples in Fortran and printed out the relative residual norm. We took  $n_1 = n_2$  and did ten random tests for each choice of  $n_1 = 2^k$  where  $k = 3, 4, \dots, 10$ . In figure 2 the relative residual norms are shown.

**6.3. The software.** For the reader who is interested in testing or evaluating our software, we implemented the routine in Fortran as well as in Matlab<sup>1</sup>. You can download it at: <http://www.cs.kuleuven.ac.be/~mase/software.html>

## References

- [1] H. Ahmed, T. Natarajan, K.R. Rao. *Discrete cosine transform*. IEEE Trans. Comput. C-23 (1974), pp. 90-93.
- [2] W. L. Briggs. *A Multigrid Tutorial*. SIAM Books, Philadelphia, 1987. First edition.
- [3] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial*. SIAM Books, Philadelphia, 2000. Second edition.
- [4] O. Buneman. *A compact non-iterative Poisson solver*. Rep. 294, Stanford University Institute for Plasma Research, Stanford, Calif., 1969.
- [5] B.L. Buzbee, G.H. Golub, C.W. Nielson. *On direct methods for solving Poisson's equation*. SIAM J. Numer. Anal. 7 (1970), pp. 627-656.
- [6] B.L. Buzbee, F.W. Dorr, J.A. George, G.H. Golub. *The direct solutions of the discrete Poisson equation on irregular regions*. SIAM J. Numer Anal., 8 (1971), pp. 722-736.
- [7] B.L. Buzbee, F.W. Dorr. *The direct solution of the biharmonic equation on rectangular regions and the Poisson equation on irregular regions*. SIAM J. Numer Anal., 11 (1974), pp. 753-763.
- [8] R.H. Chan, S. Serra Capizzano, C. Tablino-Possio *Two-grid Methods for Banded Linear Systems from DCT III Algebra*. Numer. Linear Algebra Appl., 2002.

<sup>1</sup>Matlab is a registered trademark of The Mathworks, Inc.

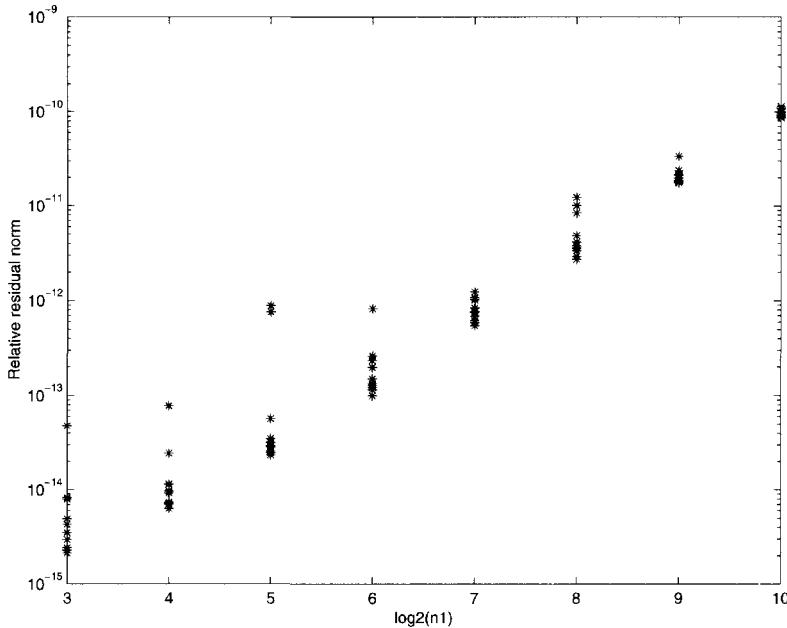


FIGURE 2. Relative residual norms for the overall algorithm applied to random examples.

- [9] R.H. Chan, M. Donatelli, S. Serra Capizzano, C. Tablino-Possio *Application of multigrid techniques to image restoration problems*. Technical report CUHK-2002-14, Department of Mathematics, Chinese University of Hong Kong, Shatin, NT, Hong Kong.
- [10] S.C. Chan, K.L. Ho. *Fast algorithms for computing the discrete cosine transform*. IEEE Trans. Circuits Syst., II, Analog Digit. Signal Process., 39 (1992), pp. 185-190.
- [11] P. Concus, G.H. Golub. *Use of fast direct methods for the efficient numerical solution of nonseparable elliptic equations*. SIAM J. Numer Anal., 10 (1973), pp. 1103-1120.
- [12] F.W. Dorr. *The direct solution of the discrete Poisson equation on a rectangle*. SIAM Rev., 12 (1970), pp.248-263.
- [13] I. Gohberg, T. Kailath, V. Olshevsky. *Fast Gaussian elimination with partial pivoting for matrices with displacement structure*. Math. Comput. 64 (1995), pp. 1557-1576.
- [14] A. Graham. *Kronecker products and matrix calculations with applications*. John Wiley, New York, 1981, Chapter 2-3.
- [15] W. Hackbusch. *Multigrid Methods and Applications*, volume 4 of *Computational Mathematics*. Springer-Verlag, Berlin, 1985.
- [16] G. Heinig, A. Bojanczyk. *Transformation techniques for Toeplitz and Toeplitz-plus-Hankel matrices. I. Transformations*. Linear Algebra Appl., 254 (1997), pp. 193-226.
- [17] G. Heinig, A. Bojanczyk. *Transformation techniques for Toeplitz and Toeplitz-plus-Hankel matrices. II. Algorithms*. Linear Algebra Appl., 278 (1998), pp. 11-36.
- [18] G. Heinig, A. Bojanczyk. *Transformation techniques for Toeplitz and Toeplitz-plus-Hankel matrices. I. Transformations*. Linear Algebra Appl., 254 (1997), pp. 193-226.
- [19] R.W. Hockney. *A fast direct solution of Poissons equation using Fourier analysis*. J. Assoc. Comput. Mach., 8 (1965), pp. 95-113.
- [20] R.W. Hockney. *The potential calculation and some applications*. Methods of Computational Physics, vol. 9, B. Adler, S. Fernback en M. Rotenberg, eds., Academic Press, New York and London, 1969, pp. 136-211.
- [21] R. W. Hockney. *POT4-A fast direct Poisson-solver for the rectangle allowing some mixed boundary conditions and internal electrodes*. Tech. report, IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598, May 1970.

- [22] J. Hendrickx. *Veralgemeende S-matrices en hun aanwendingen bij de eindigedifferentiebenaderingen van differentiaalvergelijkingen en het oplossen van symmetrische band-Toeplitzstelsels*. PhdThesis, K.U. Leuven, 2000 (in Dutch).
- [23] T. Kailath, S. Kung, M. Morf. *Displacement ranks of matrices and linear equations*. J. Math. anal. Appl. 68 (1979), pp. 395-407.
- [24] T. Kailath, A.H. Sayed. *Displacement structure: theory and applications*. SIAM Rev. 17 (1995), pp. 297-386.
- [25] T. Kailath, A.H. Sayed. *Fast reliable algorithms for matrices with structure*. SIAM, Philadelphia, 1999.
- [26] L. Mertens.  *$S_k$ -matrices en hun aanwendingen bij de eindige differentiebenaderingen van differentiaalvergelijkingen*. PhdThesis, K.U. Leuven, 1990 (in Dutch).
- [27] L. Mertens, H. Van de Vel. *A special class of structured matrices constructed with the Kronecker produkt and its use for difference equations*. Linear Algebra Appl., 106(1988), p. 117-147.
- [28] C. Van Loan. *Computational frameworks for the Fast Fourier Transform*. SIAM, Philadelphia, 1992.
- [29] M.N. Ozisik *Heat Conduction*. John Wiley, New York, 1980.
- [30] M. Pickering *An introduction to Fast Fourier Transform Methods for Partial Differential Equations, with applications*. John Wiley, New York, 1986.
- [31] W.M. Pickering and P.J. Harley *On Robbins boundary conditions, elliptic equations, and FFT methods*, J. Compt. Phys. 122, No. 2, 380-383(1995).
- [32] W.M. Pickering and P.J. Harley *FFT solution of the Robbins problem*, IMA J. Num. Anal 13, No. 2, 215-257 (1992).
- [33] W.M. Pickering and P.J. Harley *Iterative solution of the Robbins problem*. Intern. Journl. Comput. Math. 45, 243-257 ,243-257 (1992).
- [34] S. Serra Capizzano. *Convergence analysis of two grid methods for elliptic Toeplitz and PDEs Matrix-sequences*. Numerische mathematik, 92 (2002), pp. 433-465.
- [35] G. Strang. *The discrete cosine transform*. SIAM Rev., 41 (1999), pp. 135-147.
- [36] P.N. Swarztrauber, R.A. Sweet. *The direct solution of the discrete Poisson equation on a disk*. SIAM J. Numer. Anal., 10 (1973), pp. 900-907.
- [37] P.N. Swarztrauber. *The methods of cyclic reduction, fourier analysis and the FACR algorithm for the discrete solution of Poisson's equation on a rectangle*. SIAM Rev., 19 (1977),pp. 490-501.
- [38] P.N. Swarztrauber. *The direct solution of the discrete Poisson equation on the surface of a sphere*. J. Comput. Phys., 15 (1974),pp. 46-54.
- [39] P.N. Swarztrauber. *A direct method for the discrete solution of separable elliptic equations*. SIAM J. Numer. Anal., 11 (1974), pp. 1136-1150.
- [40] P.N. Swarztrauber. *Symmetric FFTs*. Math. Comput., 47 (1986), pp. 323-346.
- [41] P.N. Swarztrauber *Fast Poisson Solvers*. In MAA Studies in Mathematics, vol. 24, Studies in Numerical Analysis. Gene H. Golub, ed., The Mathematical Association of America, 1984, pp. 319-370.
- [42] R.A. Sweet *A cyclic reduction algorithm for solving block tridiagonal systems of arbitrary dimensions*. SIAM J. Numer Anal., 14 (1977), pp. 706-720.
- [43] Z. Wang, B.R. Hunt. *The discrete W transform*. Appl. Math. Comput. 16 (1985), pp. 19-48.
- [44] P. Wesseling. *An Introduction to Multigrid Methods*. John Wiley & Sons, Chichester, 1992.

DEPARTMENT OF COMPUTER SCIENCE, KATHOLIEKE UNIVERSITEIT LEUVEN, CELESTIJNENLAAN  
200 A, B-3001 LEUVEN, BELGIUM

Current address: Economische Hogeschool EHSAL, Stormstraat 2, B-1000 Brussel, Belgium  
E-mail address: jef.hendrickx@prof.ehsal.be

DEPARTMENT OF COMPUTER SCIENCE, KATHOLIEKE UNIVERSITEIT LEUVEN, CELESTIJNENLAAN  
200 A, B-3001 LEUVEN, BELGIUM

E-mail address: raf.vandebril@cs.kuleuven.ac.be

DEPARTMENT OF COMPUTER SCIENCE, KATHOLIEKE UNIVERSITEIT LEUVEN, CELESTIJNENLAAN  
200 A, B-3001 LEUVEN, BELGIUM

E-mail address: marc.vanbarel@cs.kuleuven.ac.be

# Structured Matrices in Unconstrained Minimization Methods

Carmine Di Fiore

**ABSTRACT.** Structured matrix algebras  $\mathcal{L}$  and a generalized *BFGS*-type iterative scheme have been recently exploited to introduce low complexity quasi-Newton methods, named *LQN*, for solving general (nonstructured) minimization problems. In this paper we study the *inverse LQN* methods, which define inverse Hessian approximations by an inverse *BFGS*-type updating procedure. As the known *LQN*, the inverse *LQN* methods can be implemented with only  $O(n \log n)$  arithmetic operations per step and  $O(n)$  memory allocations. Moreover, they turn out to be particularly useful in the study of conditions on  $\mathcal{L}$  which guarantee the extension of the fast *BFGS* local convergence properties to *LQN*-type algorithms.

## 1. Introduction

In solving minimization problems  $\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x})$  of large dimension  $n$ , the use of the classical Newtonian or quasi-Newtonian methods is prohibitive. For example, in the *BFGS* method [12], [21] the Hessian  $\nabla^2 f(\mathbf{x}_{k+1})$  of the Newton iterative scheme is replaced with the matrix

$$B_{k+1} = \Phi(B_k, \mathbf{s}_k, \mathbf{y}_k) = B_k + \frac{1}{\mathbf{s}_k^T \mathbf{y}_k} \mathbf{y}_k \mathbf{y}_k^T - \frac{1}{\mathbf{s}_k^T B_k \mathbf{s}_k} B_k \mathbf{s}_k \mathbf{s}_k^T B_k \quad (1.1)$$

defined in terms of the previous Hessian approximation  $B_k$  and of the difference vectors  $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$ ,  $\mathbf{y}_k = \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k)$ . The quasi-Newton iterative scheme so obtained allows to avoid the expensive Hessian evaluations and the  $O(n^3)$  complexity per iteration of the Newton method, and has a superlinear rate of convergence. However, the implementation of *BFGS* requires in general the availability of  $O(n^2)$  memory allocations and an  $O(n^2)$  amount of computation for each step.

More suitable algorithms for large scale optimization problems are the limited-memory *BFGS* (*L-BFGS*) [21] and the one step secant (*OSS-OSSV*) [1], [18], [26] methods. In both methods the idea is to define  $B_{k+1}$  in terms of only a small number of previous vectors pairs  $\mathbf{s}_j, \mathbf{y}_j$ , say  $m$  in *L-BFGS* and one, the most recent, in *OSS-OSSV*. In this way the complexity is reduced to  $O(mn)$  and  $O(n)$ , respectively. The *L-BFGS* methods have an acceptable rate of convergence, however, the optimal choice of  $m$  is problem dependent [21], [4].

Some minimization methods, named  $\mathcal{L}QN$ , have been recently introduced in [14] in order to reduce the complexity by maintaining a quasi-Newton local behaviour. The idea in  $\mathcal{L}QN$  methods is to replace (1.1) with the iterative formula  $B_{k+1} = \Phi(\mathcal{L}_{B_k}, \mathbf{s}_k, \mathbf{y}_k)$  where  $\mathcal{L}_{B_k}$  is the best least squares fit to  $B_k$  from a suitable space  $\mathcal{L}$  of matrices. If  $\mathcal{L}$  is chosen structured then the time and space complexity of  $\mathcal{L}QN$  is  $O(n \log n)$  and  $O(n)$ , respectively. Notice that in  $\mathcal{L}QN$  a structured version of the whole matrix  $B_k$  is updated, whereas, in  $L\text{-}BFGS$ , only an  $m$ -part of  $B_k$  is used in the definition of  $B_{k+1}$ . There are secant and nonsecant  $\mathcal{L}QN$  algorithms. The numerical experiences show the competitiveness of secant  $\mathcal{L}QN$  with  $L\text{-}BFGS$  and  $OSS\text{-}OSSV$  [4]. On the other side, a global linear convergence result holds for nonsecant  $\mathcal{L}QN$  methods, independently from the choice of  $\mathcal{L}$  [14], [16]. The latter result is obtained by extending the analogous  $BFGS$  convergence result of Powell [25], with the help of some crucial properties of the matrix  $\mathcal{L}_{B_k}$ .

In this paper we introduce a new class of minimization methods, named *inverse  $\mathcal{L}QN$*  ( $\mathcal{IL}QN$ ), having the same low time and space complexity of the  $\mathcal{L}QN$  algorithms considered in [14], [4], [16] (see Sections 2 and 3). They are obtained by updating inverse Hessian approximations  $H_k$  via the formula

$$H_{k+1} = \Psi(\mathcal{L}_{H_k}, \mathbf{s}_k, \mathbf{y}_k) \quad (1.2)$$

where  $\Psi$  is the operator defined by  $\Phi(B, \mathbf{s}, \mathbf{y})^{-1} = \Psi(B^{-1}, \mathbf{s}, \mathbf{y})$ . Notice that  $\mathcal{IL}QN$  differ from  $\mathcal{L}QN$ , unless  $\mathcal{L} = \mathbb{C}^{n \times n}$  in which case both coincide with  $BFGS$ .

In Sections 4 and 5 we investigate some conditions on  $\mathcal{L}$  for which the inverse  $\mathcal{L}QN$  methods have a fast local convergence. The extension of known  $BFGS$  local convergence properties (see Broyden, Dennis, Moré [7], [10], [11]) to  $\mathcal{L}QN$ -type algorithms works, in fact, better for the  $\mathcal{IL}QN$  methods rather than for the simple  $\mathcal{L}QN$ . It is shown, in particular, that if  $\mathcal{L}_{\nabla^2 f(\mathbf{x}_*)} = \nabla^2 f(\mathbf{x}_*)$ , then the  $\mathcal{IL}QN$  nonsecant version,  $\mathcal{INS}\mathcal{L}QN$ , converges superlinearly to  $\mathbf{x}_*$ , whereas the secant version,  $\mathcal{IS}\mathcal{L}QN$ , converges more than linearly. The latter result also holds if the distance between  $\mathcal{L}_{H_k}$  and  $H_k$  converges to zero as  $k \rightarrow \infty$  with such a rate that the condition (4.4) is satisfied. So, even if the numerical experiences show in any case a sufficiently fast local convergence of  $\mathcal{L}QN$ -type methods [4], a suitable choice of  $\mathcal{L}$  could result in a better performance. In particular, by observing that the space  $\mathcal{L}$  in the condition (4.4) is not necessarily fixed (i.e. the same space at each step) one deduces that a way to improve the  $\mathcal{IS}\mathcal{L}QN$  rate of convergence may consist in using, together with (1.2), an iterative procedure updating structured spaces  $\mathcal{L}$  as close as possible to  $H_{k+1}$ .

## 2. The $\mathcal{IL}QN$ minimization methods

Quasi-Newton methods for the unconstrained minimization of a function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  generate a minimizing sequence  $\{\mathbf{x}_k\} \subset \mathbb{R}^n$  by using the quasi-Newton iterative scheme  $\mathbf{x}_{k+1} = \mathbf{x}_k - \lambda_k H_k \mathbf{g}_k$ ,  $\mathbf{g}_k = \nabla f(\mathbf{x}_k)$ . The inverse Hessian approximation  $H_k$  is a real symmetric positive definite (*pd*) matrix usually defined in terms of the available second order information, rather than from scratch. In particular, in the  $BFGS$  method we have  $H_{k+1} = \Psi(H_k, \mathbf{s}_k, \mathbf{y}_k)$ ,  $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$ ,  $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$ , where  $\Psi$  is the updating operator

$$\Psi(H, \mathbf{s}, \mathbf{y}) = H - \frac{1}{\mathbf{y}^T \mathbf{s}} (H \mathbf{y} \mathbf{s}^T + \mathbf{s} \mathbf{y}^T H) + \left(1 + \frac{\mathbf{y}^T H \mathbf{y}}{\mathbf{y}^T \mathbf{s}}\right) \frac{1}{\mathbf{y}^T \mathbf{s}} \mathbf{s} \mathbf{s}^T. \quad (2.1)$$

The identity  $H_{k+1}\mathbf{y}_k = \mathbf{s}_k$  allows us to classify *BFGS* as a *secant* method, for it can be seen as a generalization of the secant method in one dimension [12]. The *BFGS* method has a fast (superlinear) local convergence and the reason is essentially in the fact that  $H_{k+1} = \Psi(H_k, \mathbf{s}_k, \mathbf{y}_k)$  with  $\Psi$  as in (2.1) retains as many information on  $H_k$  as possible. This is important to have a quasi-Newtonian behaviour in the neighbours of the solution [12].

In this paper we propose the alternative inverse Hessian updating formula

$$H_{k+1} = \Psi(\mathcal{L}_{H_k}, \mathbf{s}_k, \mathbf{y}_k) \quad (2.2)$$

where  $\mathcal{L}_{H_k}$  is the best least squares fit to  $H_k$  from a suitable space  $\mathcal{L}$  of  $n \times n$  matrices and we call  $\mathcal{ILQN}$  (inverse  $\mathcal{LQN}$ ) the corresponding Quasi-Newton methods. Obviously the same *BFGS* method can be seen as a particular  $\mathcal{ILQN}$  method and precisely as the  $\mathcal{IL}(C^{n \times n})QN$  method where  $C^{n \times n}$  is the set of all  $n \times n$  matrices with complex entries. As it is stated in the next Theorem 2.2 the  $\mathcal{ILQN}$  methods yield strictly decreasing sequences  $\{f(\mathbf{x}_k)\}$ , provided  $\mathcal{L}$  satisfies condition (2.7). Moreover, in Section 3 it will be shown that by imposing a *structure* to  $\mathcal{L}$  one can obtain  $\mathcal{ILQN}$  methods of low space and time complexity, and therefore more suitable than *BFGS* in solving large scale optimization problems. Finally, in Sections 4 and 5 some conditions on  $\mathcal{L}$  are investigated in order to extend the *BFGS* local convergence properties to  $\mathcal{ILQN}$ .

Notice that the  $\mathcal{ILQN}$  methods are different from the  $\mathcal{LQN}$  minimization algorithms studied in [14], [4], [16]. In fact, the latter exploit the updating formula  $H_{k+1} = \Psi((\mathcal{L}_{H_k^{-1}})^{-1}, \mathbf{s}_k, \mathbf{y}_k)$  which is in general not equal to (2.2).

Let  $\mathcal{L}$  be a subspace of  $C^{n \times n}$  of dimension  $m$ . The set  $C^{n \times n}$  is a Hilbert space with the inner product  $(X, Y) = \sum_{i,j=1}^n \bar{x}_{ij}y_{ij}$  and the norm induced by  $(\cdot, \cdot)$  is the Frobenius norm  $\|X\|_F = (\sum_{i,j=1}^n |x_{ij}|^2)^{1/2}$ . Thus, by the Hilbert projection theorem, there exists a unique element  $\mathcal{L}_H \in \mathcal{L}$  such that

$$\|\mathcal{L}_H - H\|_F \leq \|X - H\|_F, \quad \forall X \in \mathcal{L}, \quad (2.3)$$

or, equivalently, such that

$$(H - \mathcal{L}_H, X) = 0, \quad \forall X \in \mathcal{L}. \quad (2.4)$$

The matrix  $\mathcal{L}_H$  is called *the best least squares (l.s.) fit to  $H$  from  $\mathcal{L}$*  [9], [17], [23]. The orthogonality condition (2.4) allows us to give an explicit form to  $\mathcal{L}_H$ :

**Proposition 2.1.** *If  $\{J_1, J_2, \dots, J_m\}$  is a basis of  $\mathcal{L}$ , then*

$$\mathcal{L}_H = \sum_{k=1}^m [W^{-1}\mathbf{c}]_k J_k, \quad w_{ij} = (J_i, J_j), \quad c_i = (J_i, H). \quad (2.5)$$

As a consequence, we have:

- i)  $\mathcal{L}_H$  is linear, i.e.  $\mathcal{L}_{\alpha X + \beta Y} = \alpha \mathcal{L}_X + \beta \mathcal{L}_Y$ ;
- ii) if the  $J_k$  are real, then  $\mathcal{L}_H$  is real whenever  $H$  is real.

The  $\mathcal{ILQN}$  methods, in their secant ( $\mathcal{IS LQN}$ ) and nonsecant ( $\mathcal{INS LQN}$ ) versions, are defined as follows

$$\mathbf{x}_0 \in \mathbb{R}^n, H_0 = pd \text{ } n \times n \text{ matrix.}$$

For  $k = 0, 1, \dots$ :

$$\begin{cases} \mathbf{d}_k = \begin{cases} -H_k \mathbf{g}_k & \mathcal{IS} \\ -\mathcal{L}_{H_k} \mathbf{g}_k & \mathcal{INS} \end{cases} \\ \mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_k \mathbf{d}_k \\ \mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k, \mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k \\ H_{k+1} = \Psi(\mathcal{L}_{H_k}, \mathbf{s}_k, \mathbf{y}_k) \end{cases} \quad (2.6)$$

where  $\mathcal{L}$  is assumed to verify the implication

$$H_k pd \Rightarrow \mathcal{L}_{H_k} pd. \quad (2.7)$$

Notice that in  $\mathcal{INS}$  the matrix  $H_{k+1}$  solves the secant equation  $X\mathbf{y}_k = \mathbf{s}_k$ , but  $\mathbf{d}_{k+1}$  is not defined directly in terms of  $H_{k+1}$ . So only  $\mathcal{IS LQN}$  is a secant method like *BFGS*.

Both  $\mathcal{IS}$  and  $\mathcal{INS}$  yield descent directions  $\mathbf{d}_{k+1}$ . In fact, since  $\Psi(\cdot, \mathbf{s}, \mathbf{y})$  maps  $pd$  matrices into  $pd$  matrices whenever the inner product  $\mathbf{s}^T \mathbf{y}$  is positive [12], by (2.7) we have

$$H_k pd \Rightarrow \begin{cases} \mathcal{L}_{H_k} pd \\ \mathbf{s}_k^T \mathbf{y}_k > 0 \end{cases} \Rightarrow H_{k+1} pd \Rightarrow \mathcal{L}_{H_{k+1}} pd. \quad (2.8)$$

Theorem 2.2 below shows that the condition  $\mathbf{s}_k^T \mathbf{y}_k > 0$  can be obtained by a suitable choice of  $\lambda_k$ .

**Theorem 2.2.** *Let  $f$  be continuously differentiable and lower bounded. If the subspace  $\mathcal{L}$  of  $\mathbb{C}^{n \times n}$  satisfies the condition (2.7) and if the step-length  $\lambda_k$  belongs to the Armijo-Goldstein set  $\Lambda_k$ ,*

$$\Lambda_k = \{\lambda \in \mathbb{R}^+ : \chi_k(\lambda) \leq \chi_k(0) + \lambda c_1 \chi'_k(0) \text{ and } \chi'_k(\lambda) \geq c_2 \chi'_k(0)\}$$

where  $\chi_k(\lambda) = f(\mathbf{x}_k + \lambda \mathbf{d}_k)$ ,  $0 < c_1 < c_2 < 1$ , then both the  $\mathcal{IS}$  and the  $\mathcal{INS LQN}$  algorithms generate real sequences  $\{\mathbf{x}_k\}$ ,  $\{\mathbf{s}_k^T \mathbf{y}_k\}$  and  $\{f(\mathbf{x}_k)\}$  with  $\mathbf{s}_k^T \mathbf{y}_k > 0$  and  $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$  for every  $k$ .

**PROOF.** It is sufficient to note that if  $\chi'_k(0) = \mathbf{g}_k^T \mathbf{d}_k < 0$  then the *AG* set  $\Lambda_k$  is nonempty and that the choice  $\lambda_k \in \Lambda_k$  yields the inequalities  $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$  and  $\mathbf{s}_k^T \mathbf{y}_k > 0$ . So, by (2.8), one has  $\chi'_{k+1}(0) = \mathbf{g}_{k+1}^T \mathbf{d}_{k+1} < 0$  unless  $\mathbf{g}_{k+1} = \mathbf{0}$ .  $\square$

The condition (2.7) turns out to be verified for spaces  $\mathcal{L}$  diagonalized by unitary matrices  $U_{\mathcal{L}}$  ( $\mathcal{L} = SDU_{\mathcal{L}}$ ) and spanned by  $n$  real matrices. The corresponding  $\mathcal{I} (SDU_{\mathcal{L}})QN$  methods are studied in detail in the following Section 3. Notice that the same *BFGS* method can be seen as a  $\mathcal{I} (SDU_{\mathcal{L}})QN$  method where  $U_{\mathcal{L}}$  is changed at each step and is precisely chosen equal to the orthonormal matrix diagonalizing  $H_k$  (so that  $\mathcal{L}_{H_k} = H_k$ ).

### 3. $\mathcal{I} (SDU_{\mathcal{L}})QN$ methods

In this section it is shown that if  $\mathcal{L} = SDU_{\mathcal{L}}$  then each step of the  $\mathcal{IS}$  and  $\mathcal{INS LQN}$  algorithms (2.6) can be performed by computing a small number of vector inner products and two linear transforms involving the matrix  $U_{\mathcal{L}}$ . Thus, if  $U_{\mathcal{L}}$  is the Fourier matrix  $F = \frac{1}{\sqrt{n}}(e^{-i2\pi ij/n})_{i,j=0}^{n-1}$  or any trigonometric or Hartley-type matrix reported in [3], [5], [6], [19], [20], [23], [29], [30], defining a fast transform,

then  $\mathcal{IS}$  and  $\mathcal{INS}$  can be implemented with only  $O(n \log n)$  flops per step and  $O(n)$  memory allocations. So, by imposing a *structure* to the space  $\mathcal{L}$ , a strong reduction of time and space complexity with respect to *BFGS* is obtained. The structure is obviously emphasized in the Fourier case where  $\mathcal{L}$  represents the space  $\mathcal{C}$  of circulant matrices  $(c_{i-j \bmod n})_{i,j=0}^{n-1}$ ,  $c_k \in \mathbb{C}$ . However, some different, more hidden structures could yield algorithms of the same or even better computational efficiency (see Section 5).

At the end of this section, the inverse  $(SDU_{\mathcal{L}})QN$  methods corresponding to two different particular choices of  $\mathcal{L}$  are exploited to minimize a simple function taken from [12]. The choice for which  $\mathcal{L}_{\nabla^2 f(\mathbf{x}_*)} = \nabla^2 f(\mathbf{x}_*)$  leads to better results and this experimental remark introduces to the  $\mathcal{ILQN}$  local convergence study of Sections 4 and 5.

Let  $U_{\mathcal{L}}$  be a  $n \times n$  unitary matrix,  $U_{\mathcal{L}}^* = U_{\mathcal{L}}^{-1}$ , and set

$$\mathcal{L} = SDU_{\mathcal{L}} = \{U_{\mathcal{L}}d(\mathbf{z})U_{\mathcal{L}}^* : \mathbf{z} \in \mathbb{C}^n\} \quad (3.1)$$

where  $d(\mathbf{z}) = \text{diag}(z_i, i = 1, \dots, n)$ ,  $\mathbf{z} = [z_1 z_2 \cdots z_n]^T$ . As  $\mathcal{L}$  is a subspace of  $\mathbb{C}^{n \times n}$ , for an arbitrary matrix  $H \in \mathbb{C}^{n \times n}$  the best l.s. fit to  $H$  from  $\mathcal{L}$ ,  $\mathcal{L}_H$ , is well defined. One can state the following known properties of  $\mathcal{L}_H$ .

**Proposition 3.1.** *i)  $\mathcal{L}_H = U d(\mathbf{z}_H) U^*$  where  $[\mathbf{z}_H]_j = [U^* H U]_{jj}$ . In particular  $\mathbf{z}_{\mathbf{x}\mathbf{y}^T} = d(U^* \mathbf{x}) U^T \mathbf{y}$ ,  $\mathbf{x}, \mathbf{y} \in \mathbb{C}^n$ .*

*ii) If  $H = H^*$ , then  $\mathcal{L}_H = \mathcal{L}_H^*$  and  $\min \nu(H) \leq \nu(\mathcal{L}_H) \leq \max \nu(H)$  where  $\nu(X)$  denotes the generic eigenvalue of  $X$ . Therefore  $\mathcal{L}_H$  is hermitian positive definite whenever  $H$  is hermitian positive definite.*

*iii) If  $\mathcal{L}$  is spanned by real matrices then  $\mathcal{L}_H$  is pd whenever  $H$  is pd.*

**PROOF.** The proof of i) lies in the following equality

$$\|U d(\mathbf{z}) U^* - H\|_F = \|d(\mathbf{z}) - U^* H U\|_F.$$

The properties in ii) can be obtained directly from i). Finally ii) and Proposition 2.1,ii), imply iii).  $\square$

As an immediate consequence of Proposition 3.1, iii), the conclusions of Theorem 2.2 hold for  $\mathcal{L} = SDU_{\mathcal{L}}$  provided that  $SDU_{\mathcal{L}}$  is spanned by real matrices. This will be assumed in the following, referring of  $SDU_{\mathcal{L}}$  spaces. So if  $\lambda_k \in \Lambda_k$  both the  $\mathcal{IS}$  and the  $\mathcal{INS}$   $(SDU_{\mathcal{L}})QN$  algorithms generate real sequences  $s_k^T \mathbf{y}_k > 0$  and  $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$ .

The property i) of  $\mathcal{L}_H$  is used to prove the next Theorem 3.2 concerning the complexity of  $\mathcal{ILQN}$ . In particular, we obtain the identities (3.3)–(3.5) which imply that the  $\mathcal{ILQN}$  search direction can be defined in terms of the only eigenvalues of  $\mathcal{L}_{H_k}$ . Notice that the spectrum of  $\mathcal{L}_{H_k}$  is strictly related to the spectrum of the original inverse Hessian approximation  $H_k$  since we have

$$\begin{aligned} \nu_1(H_k) + \cdots + \nu_i(H_k) &\leq \nu_1(\mathcal{L}_{H_k}) + \cdots + \nu_i(\mathcal{L}_{H_k}), \\ \nu_i(\mathcal{L}_{H_k}) + \cdots + \nu_n(\mathcal{L}_{H_k}) &\leq \nu_i(H_k) + \cdots + \nu_n(H_k) \end{aligned} \quad (3.2)$$

where  $\nu_i(X)$  are the eigenvalues of  $X$  in nondecreasing order [27], [28]. It is by this reason that the  $\mathcal{ILQN}$  seems to be very close to the *BFGS* search direction.

**Theorem 3.2.** *The  $\mathcal{IS}$  and  $\mathcal{INS}$   $\mathcal{LQN}$  methods (2.6) with  $\mathcal{L} = SDU_{\mathcal{L}}$ , require at each step the computation of two  $U_{\mathcal{L}}$ -discrete transforms plus  $O(n)$  flops. Thus*

their time complexity and space complexity are  $nO(n \log n)$  and  $O(n)$ , respectively, whenever  $U_{\mathcal{L}}$  defines a fast transform.

**PROOF.** The result can be achieved by rewriting the algorithm (2.6) in terms of the matrix  $U = U_{\mathcal{L}}$ . By the linearity of  $\mathcal{L}_H$  and by the definition of  $\mathbf{z}_H$  in Proposition 3.1, the updating formula in (2.2) yields

$$\mathbf{z}_{H_{k+1}} = \mathbf{z}_{H_k} - \frac{1}{\mathbf{y}_k^T \mathbf{s}_k} \left( \mathbf{z}_{\mathcal{L}_{H_k} \mathbf{y}_k \mathbf{s}_k^T} + \mathbf{z}_{\mathbf{s}_k (\mathcal{L}_{H_k})^T \mathbf{y}_k} \right) + \left( 1 + \frac{\mathbf{y}_k^T \mathcal{L}_{H_k} \mathbf{y}_k}{\mathbf{y}_k^T \mathbf{s}_k} \right) \frac{1}{\mathbf{y}_k^T \mathbf{s}_k} \mathbf{z}_{\mathbf{s}_k \mathbf{s}_k^T}.$$

By Proposition 3.1 this formula can be rewritten as follows:

$$\mathbf{z}_{H_{k+1}} = \mathbf{z}_{H_k} - \frac{1}{\mathbf{y}_k^T \mathbf{s}_k} \left[ d(\mathbf{z}_{H_k}) 2 \operatorname{Re}(d(U^* \mathbf{y}_k) U^T \mathbf{s}_k) - \left( 1 + \frac{\mathbf{z}_{H_k}^T |U^* \mathbf{y}_k|^2}{\mathbf{y}_k^T \mathbf{s}_k} \right) |U^* \mathbf{s}_k|^2 \right] \quad (3.3)$$

where  $|\mathbf{z}|^2$  is the vector whose  $i$ th entry is  $|z_i|^2$ . Moreover the following formulas for the search directions  $\mathbf{d}_{k+1}$ , respectively, in  $\mathcal{IS}$  and  $\mathcal{INS}$  hold:

$$\begin{aligned} U^* \mathbf{d}_{k+1} &= -d(\mathbf{z}_{H_k}) U^* \mathbf{g}_{k+1} + \frac{\mathbf{s}_k^T \mathbf{g}_{k+1}}{\mathbf{y}_k^T \mathbf{s}_k} d(\mathbf{z}_{H_k}) U^* \mathbf{y}_k \\ &\quad + \left[ -\left( 1 + \frac{(\mathbf{z}_{H_k})^T |U^* \mathbf{y}_k|^2}{\mathbf{y}_k^T \mathbf{s}_k} \right) \frac{\mathbf{s}_k^T \mathbf{g}_{k+1}}{\mathbf{y}_k^T \mathbf{s}_k} \right. \\ &\quad \left. + \frac{(\mathbf{z}_{H_k})^T d(U^T \mathbf{y}_k) U^* \mathbf{g}_{k+1}}{\mathbf{y}_k^T \mathbf{s}_k} \right] U^* \mathbf{s}_k. \end{aligned} \quad (3.4)$$

$$U^* \mathbf{d}_{k+1} = -d(\mathbf{z}_{H_{k+1}}) U^* \mathbf{g}_{k+1}. \quad (3.5)$$

Here below the algorithm (2.6) is rewritten using (3.3), (3.4), (3.5).

$$\mathbf{x}_0 \in \mathbb{R}^n, H_0 = pd \text{ } n \times n \text{ matrix},$$

$$U^* \mathbf{d}_0 = \begin{cases} -U^* H_0 \mathbf{g}_0 & \mathcal{IS} \\ -d(\mathbf{z}_{H_0}) U^* \mathbf{g}_0 & \mathcal{INS} \end{cases}.$$

$$\mathbf{d}_0 = U(U^* \mathbf{d}_0).$$

$$\text{For } k = 0, 1, \dots :$$

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_k \mathbf{d}_k \\ \mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k, \quad \mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k \\ (3.3) \\ \begin{cases} (3.4) & \mathcal{IS} \\ (3.5) & \mathcal{INS} \end{cases} \\ \mathbf{d}_{k+1} = U(U^* \mathbf{d}_{k+1}) \end{cases} \quad (3.6)$$

From (3.6) it is clear that each step of the  $\mathcal{ILQN}$  methods (2.6) consists in computing the two transforms  $U^* \cdot \mathbf{g}_{k+1}$  and  $U \cdot (U^* \mathbf{d}_{k+1})$  (in  $\mathcal{IS}$ ,  $U^* \mathbf{d}_{k+1}$  can be computed from  $U^* \mathbf{s}_k = \lambda_k U^* \mathbf{d}_k$ ), and in performing  $O(n)$  flops.  $\square$

Notice that formulas (3.4) and (3.5) differ from the formulas defining the  $\mathcal{S}$  and the  $\mathcal{NS}$  search directions in  $\mathcal{LQN}$  methods (see [14], [4]). In particular, the eigenvalues  $(\mathbf{z}_{H_k})_i$ , appearing in formula (3.4), do not coincide with the inverses of the eigenvalues  $(\mathbf{z}_{B_k})_i$ , involved in the  $\mathcal{S}$   $\mathcal{LQN}$  search direction formula, as  $\mathcal{L}_{H_k} \neq \mathcal{L}_{B_k}^{-1} = (\mathcal{L}_{H_k^{-1}})^{-1}$ .

We have applied the  $\mathcal{IS}$   $\mathcal{LQN}$  minimization algorithm (3.6) with  $H_0 = I$  to the Rosenbrock and Powell test functions [12], by setting, respectively,  $\mathbf{x}_0 = [-1.2 \ 1 \ \dots]^T$  and  $\mathbf{x}_0 = [3 \ -1 \ 0 \ 1 \ \dots]^T$ . The steplength  $\lambda_k$  is chosen in the Armijo-Goldstein set  $\Lambda_k$ , by using a line-search technique of Dennis-Schnabel-type [12], the

same exploited to implement  $\mathcal{L}QN$  in [4]. The space  $\mathcal{L}$  is the matrix algebra  $\mathcal{H}$  studied in [2]:

$$\mathcal{H} = SDU_{\mathcal{H}}, \quad [U_{\mathcal{H}}]_{ij} = \frac{1}{\sqrt{n}} \left( \cos \frac{2\pi ij}{n} + \sin \frac{2\pi ij}{n} \right).$$

The matrix  $U_{\mathcal{H}}$  defines the discrete Hartley transform, which is a well known real transform computable in  $O(n \log n)$  flops [3], [6], [19], [29].

$f$	$BFGS$	$\mathcal{IS}$	$\mathcal{HQN}$	$\epsilon$
Rosenbrock	7	5	$10^{-1}$	
$n = 2$	17	13	$10^{-7}$	
Rosenbrock	26	9	$10^{-1}$	
$n = 64$	80	14	$10^{-7}$	
Rosenbrock	30	10	$10^{-1}$	
$n = 512$	85(67s)	14(0s)	$10^{-7}$	
Powell $n = 4$	12	26	$10^{-2}$	
Powell $n = 64$	30	22	$10^{-2}$	

Table 1.  $k$ :  $f(\mathbf{x}_k) < \epsilon$ ;  $s$  =seconds.

The competitiveness of  $\mathcal{ILQN}$  methods with  $BFGS$  is already clear in the preliminary experiences reported in Table 1 (machine precision= $2 \cdot 10^{-16}$ ). Notice that one could exploit the block diagonal form of the Hessian in Rosenbrock and Powell functions to improve the total efficiency of  $BFGS$ , by reducing allocation memory [12], [21]. But also the  $\mathcal{ILQN}$  total efficiency could be improved, for instance by approximating with  $\mathcal{L}$  matrices the Hessian blocks instead of the Hessian. We stress that in the implementation of  $\mathcal{ILQN}$  methods we do not need to look for some sparsity or structure in the Hessian, because, *for any Hessian*, they require only  $O(n \log n)$  flops per iteration and  $O(n)$  memory allocations. Also notice that in some important applications, as in learning neural networks [4], it is not usual to have a sparse or structured Hessian. Other runnings of  $\mathcal{IS} \mathcal{HQN}$  have shown that the line search technique exploited in the  $\mathcal{LQN}$  implementation (see [4]) does not work always efficiently for  $\mathcal{ILQN}$ . However, as the following numerical example shows, further investigations to define a line search procedure suitable for  $\mathcal{ILQN}$  are fully justified.

The data in the Table 2 refers to a simple test function found in [12]:  $f(\mathbf{x}) = x_1^4 + (x_1 + x_2)^2 + (e^{x_2} - 1)^2$ . Here  $\mathbf{x}_0 = [1 \ 1]^T$ ,  $H_0 = I$  and  $\lambda_k = 1, \forall k$ . The working machine precision is  $4.3 \cdot 10^{-19}$ .

$k$	$BFGS$	$\mathcal{IS} \mathcal{HQN}$	$\mathcal{IS} \Gamma QN$	$\mathcal{INS} \mathcal{HQN}$	$\mathcal{INS} \Gamma QN$
4	.026	.026	.025	.18	.202
10	.00012	.000074	$7.2 \cdot 10^{-6}$	.0053	$4.46 \cdot 10^{-9}$
12	$3.2 \cdot 10^{-7}$	$6.9 \cdot 10^{-7}$	$1.9 \cdot 10^{-9}$	.0013	$7.9 \cdot 10^{-14}$
14	$3.7 \cdot 10^{-11}$	$8.1 \cdot 10^{-8}$	$3.3 \cdot 10^{-14}$	.000034	
15	$1.9 \cdot 10^{-13}$	$1.09 \cdot 10^{-8}$	$5.06 \cdot 10^{-17}$	.000056	
16	$3.8 \cdot 10^{-16}$	$2.7 \cdot 10^{-10}$		.000046	

Table 2.  $\|\mathbf{x}_k - \mathbf{x}_*\|_\infty$  when  $\nabla^2 f(\mathbf{x}_*) \in \Gamma$ .

Table 2 shows that the choice of a suitable space  $\mathcal{L}$  can improve the performance of  $\mathcal{ILQN}$ . The spaces  $\mathcal{L}$  considered are, respectively,  $\mathcal{L} = \mathcal{H}$  and  $\mathcal{L} = \Gamma$  where, for

$n = 2$ ,

$$\mathcal{H} = \begin{bmatrix} a & b \\ b & a \end{bmatrix}, \quad \Gamma = \begin{bmatrix} a & b \\ b & a+b \end{bmatrix}.$$

It is clear that the second choice of  $\mathcal{L}$  leads to better results. Moreover, the  $\mathcal{I}\Gamma QN$  methods outperform the  $BFGS$  method. Notice also that in this example the nonsecant version of  $\mathcal{I}\Gamma QN$  outperforms the secant one, whereas, in general, secant  $LQN$ -type algorithms are quite superior than nonsecant. The reason of the above results is in the fact that

$$\nabla^2 f(\mathbf{x}_*) = \begin{bmatrix} 2 & 2 \\ 2 & 4 \end{bmatrix} \in \Gamma$$

and the next section gives a theoretical justification of this remark.

#### 4. Local convergence results

Let  $\mathbf{x}_* \in \mathbb{R}^n$  be a strong local minimum for  $f$ , that is  $\nabla f(\mathbf{x}_*) = \mathbf{0}$  and  $H_* = \nabla^2 f(\mathbf{x}_*)^{-1}$  is *pd*. Let  $\|\cdot\|$  denote both the euclidean vector norm and the corresponding induced spectral matrix norm. Assume that  $\mathbf{g} = \nabla f$  has continuous derivatives in an open convex set  $\mathcal{D}$  including  $\mathbf{x}_*$ , and  $\|\nabla^2 f(\mathbf{x}) - \nabla^2 f(\mathbf{x}_*)\| \leq l\|\mathbf{x} - \mathbf{x}_*\|$ ,  $l \geq 0, \forall \mathbf{x} \in \mathcal{D}$ . Under these conditions Broyden, Dennis and Moré [7] obtained the following crucial result in proving the  $BFGS$  local superlinear rate of convergence (see also [24]):

**BDM.** *For any  $r \in (0, 1)$ , there exist  $\epsilon_1(r), \delta(r) > 0$  such that the approximations  $\mathbf{x}_k$  generated by the unmodified  $BFGS$  algorithm (i.e. (2.6) with  $\mathcal{L} = \mathbb{C}^{n \times n}$  and  $\lambda_k = 1$ ) are well defined and satisfy the inequality*

$$\|\mathbf{x}_{k+1} - \mathbf{x}_*\| < r\|\mathbf{x}_k - \mathbf{x}_*\|, \quad k = 0, 1, \dots, \quad (4.1)$$

*provided that  $\|\mathbf{x}_0 - \mathbf{x}_*\| < \epsilon_1$  and  $\|H_0 - H_*\| < \delta$ . So the  $BFGS$  sequence  $\{\mathbf{x}_k\}$  converges linearly to  $\mathbf{x}_*$ , for all  $r \in (0, 1)$ .*

In Theorem 4.3, following the same steps in [7], [10], we extend the BDM result to  $\mathcal{IS} LQN$  algorithms (2.6) satisfying a condition on the sequence  $\{\mathcal{L}_{H_k}\}$  (see (4.4)) which is slightly weaker than  $\mathcal{L} = \mathbb{C}^{n \times n}$  ( $\mathcal{L}$  in Theorem 4.3 is not necessarily equal to  $SDU_{\mathcal{L}}$ ). We also prove, in Theorem 4.5, that the same BDM result can be obtained for both the  $\mathcal{IS}$  and the  $\mathcal{INS} LQN$  methods by assuming  $\mathcal{L} = SDU_{\mathcal{L}}$  where  $U_{\mathcal{L}}$  is the same unitary matrix diagonalizing  $\nabla^2 f(\mathbf{x}_*)$ . Moreover, for such particular choice of  $\mathcal{L}$ , the  $\mathcal{INS}$  version converges superlinearly.

Let us first state some auxiliary results, consisting in the inequality (4.3) and in Lemmas 4.1 and 4.2. From the updating formula (2.2) it follows that, for any real symmetric matrix  $H$ , we have

$$H_{k+1} - H = P_k(\mathcal{L}_{H_k} - H)P_k^T + [(\mathbf{s}_k - H\mathbf{y}_k)\mathbf{s}_k^T + \mathbf{s}_k(\mathbf{s}_k - H\mathbf{y}_k)^T P_k^T] \frac{1}{\mathbf{y}_k^T \mathbf{s}_k} \quad (4.2)$$

where  $P_k = I - \frac{\mathbf{s}_k \mathbf{y}_k^T}{\mathbf{y}_k^T \mathbf{s}_k}$ . For  $H = H_*$ , the identity (4.2) yields the inequality

$$\|H_{k+1} - H_*\|_{M^{-1}} \leq \frac{1}{\omega_{k,M}^2} \|\mathcal{L}_{H_k} - H_*\|_{M^{-1}} + \frac{2}{\omega_{k,M}^2} \frac{\|M^{-1}\mathbf{s}_k - M\mathbf{y}_k\|}{\|M\mathbf{y}_k\|} \quad (4.3)$$

where the matrix  $M$  is fixed real symmetric such that  $M^2 = H_*$ , the pseudo norm  $\|\cdot\|_{M^{-1}}$  is defined by  $\|X\|_{M^{-1}} = \|M^{-1}XM^{-1}\|_F$  and

$$\omega_{k,M} = \frac{\mathbf{y}_k^T \mathbf{s}_k}{\|M^{-1} \mathbf{s}_k\| \|M \mathbf{y}_k\|}.$$

Formulas (4.2) and (4.3) are obtained by proceeding as in the case  $\mathcal{L}_{H_k} = H_k$  considered in [10] (actually (4.3) holds for any *pd* matrix  $H$  in place of  $H_*$ ).

**Lemma 4.1 ([7],[22]).** *We have*

$$\|\mathbf{g}(\mathbf{v}) - \mathbf{g}(\mathbf{u}) - \nabla^2 f(\mathbf{x}_*)(\mathbf{v} - \mathbf{u})\| \leq l\sigma(\mathbf{u}, \mathbf{v})\|\mathbf{v} - \mathbf{u}\|, \quad \forall \mathbf{u}, \mathbf{v} \in \mathcal{D},$$

where  $\sigma(\mathbf{u}, \mathbf{v}) = \max\{\|\mathbf{u} - \mathbf{x}_*\|, \|\mathbf{v} - \mathbf{x}_*\|\}$ . Moreover, there exist  $\epsilon_2, \epsilon_3 > 0$  such that if  $\sigma(\mathbf{u}, \mathbf{v}) \leq \epsilon_3$ , then  $\mathbf{u}, \mathbf{v} \in \mathcal{D}$  and

$$\epsilon_2\|\mathbf{v} - \mathbf{u}\| \leq \|\mathbf{g}(\mathbf{v}) - \mathbf{g}(\mathbf{u})\| \leq \frac{1}{\epsilon_2}\|\mathbf{v} - \mathbf{u}\|.$$

**Lemma 4.2 ([10]).** *If  $\|\mathbf{u} - \mathbf{v}\| \leq \alpha\|\mathbf{u}\|$  for some  $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ ,  $\mathbf{u}, \mathbf{v} \neq \mathbf{0}$ ,  $\alpha \in \mathbb{R}$ , then  $1 - (\mathbf{u}^T \mathbf{v} / \|\mathbf{u}\| \|\mathbf{v}\|)^2 \leq \alpha^2$ . Moreover, if  $\alpha \leq 1$ , then  $\mathbf{u}^T \mathbf{v} > 0$ .*

Now the following three inequalities (A),(B),(C) are a key step for proving the desired convergence results, stated in the next Theorems 4.3 and 4.5:

$$\|\mathcal{L}_{H_{j+1}} - H_*\|_{M^{-1}} \leq \|H_{j+1} - H_*\|_{M^{-1}}, \quad (\text{A})$$

$$\|H_{j+1} - H_*\|_{M^{-1}} \leq [1 + \gamma_1 \sigma_{j,j+1}] \|\mathcal{L}_{H_j} - H_*\|_{M^{-1}} + \gamma_2 \sigma_{j,j+1}, \quad (\text{B})$$

$$[1 + \gamma_1 \sigma_{j,j+1}] \|\mathcal{L}_{H_j} - H_*\|_{M^{-1}} + \gamma_2 \sigma_{j,j+1} \leq [1 + \gamma_1 \sigma_{j,j+1}] \|H_j - H_*\|_{M^{-1}} + \gamma_2 \sigma_{j,j+1} \quad (\text{C})$$

where  $\gamma_1 = \|H_*\|l$ ,  $\gamma_2 = 4\|M\|\|M^{-1}\|l/\epsilon_2$  and  $\sigma_{i,j} = \sigma(\mathbf{x}_i, \mathbf{x}_j)$ . In Theorem 4.3 the inequalities (A) and (C) are not necessarily verified; (A) and (C) are in fact replaced by the condition (4.4). On the contrary, in Theorem 4.5 the matrices  $\mathcal{L}_{H_j}$  satisfy all the inequalities (A), (B) and (C), provided that  $H_* = \mathcal{L}_{H_*}$ .

Define  $\rho, \mu, \eta$  by the identities  $\rho = \|H_*^{-1}\|$ ,  $\mu = \|H_*\|$ ,  $\|H\| \leq \eta\|H\|_{M^{-1}}$ .

**Theorem 4.3.** *Let  $r$ ,  $0 < r < 1$ , be fixed. There exist  $\epsilon_1 = \epsilon_1(r), \delta = \delta(r) > 0$  such that, if:*

- $\|\mathbf{x}_0 - \mathbf{x}_*\| < \epsilon_1$ ,
- $H_0$  is *pd* and  $\|H_0 - H_*\|_{M^{-1}} < \delta$ ,
- $\mathbf{x}_{j+1} = \mathbf{x}_j - H_j \mathbf{g}_j$  and  $H_{j+1} = \Psi(\mathcal{L}_{H_j}, \mathbf{s}_j, \mathbf{y}_j)$ ,
- $\mathcal{L}_{H_j}$  are *pd* matrices and

$$\sum_{j=0}^{+\infty} \|\mathcal{L}_{H_j} - H_j\|_{M^{-1}} < \Omega < \frac{r}{\eta\rho}, \quad (4.4)$$

then, for  $j = 0, 1, \dots$ , we have

1.  $\|H_j\| < \eta(\delta + \Omega) + \mu$  and  $\|H_j^{-1}\| < \frac{\rho}{1-r}$ ;
2. If  $\mathbf{x}_j \neq \mathbf{x}_*$ , then

$$\|\mathbf{x}_{j+1} - \mathbf{x}_*\| < r\|\mathbf{x}_j - \mathbf{x}_*\| < \epsilon_1, \quad (4.5)$$

otherwise the algorithm stops at step  $j$ ;

3.  $\mathbf{y}_j^T \mathbf{s}_j > 0$  and therefore  $H_{j+1}$  is a well defined *pd* matrix;
4. (B) holds and then  $\|H_{j+1} - H_*\|_{M^{-1}} < \delta + \Omega$ .

Thus the BDM convergence result (4.1) holds for the IS LQN sequence  $\{\mathbf{x}_k\}$  provided that (4.4) is verified. The convergence of BFGS is retrieved for  $\mathcal{L}_{H_j} = H_j$ .

PROOF. For  $j = 0$  the proof of 1.-4. is left to the reader (follow the next steps 1.-4. with  $k = 0$ ). Assume that 1.-4. hold for  $j = 0, 1, \dots, k - 1$  and prove them for  $j = k$ .

$$1. \|H_k\| \leq \|H_k - H_*\| + \|H_*\| \leq \eta\|H_k - H_*\|_{M^{-1}} + \mu < \eta(\delta + \Omega) + \mu.$$

If  $\rho\eta(\delta + \Omega) \leq r$ , then

$$\begin{aligned} \|H_k^{-1}\| &\leq \|H_k^{-1}H_*\|\|H_*^{-1}\| = \|(I + H_*^{-1}(H_k - H_*))^{-1}\|\rho \\ &\leq \frac{\rho}{1 - \|H_*^{-1}(H_k - H_*)\|} < \frac{\rho}{1 - \rho\eta(\delta + \Omega)} \leq \frac{\rho}{1 - r}. \end{aligned}$$

2. If  $\mathbf{x}_k \neq \mathbf{x}_*$ , then, by Lemma 4.1,  $H_k \mathbf{g}_k \neq \mathbf{0}$  and

$$\begin{aligned} \|\mathbf{x}_{k+1} - \mathbf{x}_*\| &= \|\mathbf{x}_k - H_k \mathbf{g}_k - \mathbf{x}_*\| \\ &= \|(I - H_k H_*^{-1})(\mathbf{x}_k - \mathbf{x}_*) - H_k(\mathbf{g}_k - \mathbf{g}(\mathbf{x}_*) - H_*^{-1}(\mathbf{x}_k - \mathbf{x}_*))\| \\ &\leq [\rho\|H_k - H_*\| + \|H_k\|l\sigma(\mathbf{x}_k, \mathbf{x}_*)]\|\mathbf{x}_k - \mathbf{x}_*\| \\ &< [\rho\eta(\delta + \Omega) + (\eta(\delta + \Omega) + \mu)l\epsilon_1]\|\mathbf{x}_k - \mathbf{x}_*\| \\ &\leq r\|\mathbf{x}_k - \mathbf{x}_*\| \end{aligned}$$

provided that

$$\epsilon_1 \leq \epsilon_3 \quad (4.6)$$

and  $\rho\eta(\delta + \Omega) + (\eta(\delta + \Omega) + \mu)l\epsilon_1 \leq r$ . In particular  $\|\mathbf{x}_{k+1} - \mathbf{x}_*\| < \epsilon_1$ .

3. Since  $\sigma_{k,k+1} \leq \epsilon_3$ , one obtains

$$H_k \mathbf{g}_k \neq \mathbf{0} \Rightarrow \mathbf{s}_k \neq \mathbf{0} \Rightarrow \mathbf{y}_k \neq \mathbf{0}.$$

Moreover

$$\|M^{-1}\mathbf{s}_k - M\mathbf{y}_k\| \leq \frac{\|H_*\|\|H_*^{-1}\mathbf{s}_k - \mathbf{y}_k\|}{\|\mathbf{s}_k\|} \|M^{-1}\mathbf{s}_k\| \leq \alpha_k \|M^{-1}\mathbf{s}_k\|,$$

$\alpha_k = \mu l\sigma_{k,k+1}$ . Thus, by Lemma 4.2,  $1 - \omega_{k,M}^2 \leq \alpha_k^2$  where  $\alpha_k \leq 1$  if, for example,  $\mu l\epsilon_1 < \frac{1}{2}$ . In this case  $\mathbf{y}_k^T \mathbf{s}_k > 0$ , i.e. the matrix  $H_{k+1}$  in (2.2) is well defined and  $pd$  (see (2.8)).

4. Then  $H_{k+1}$  satisfies (4.3) where, by (4.6),

$$\frac{\|M^{-1}\mathbf{s}_k - M\mathbf{y}_k\|}{\|M\mathbf{y}_k\|} \leq \frac{\|M\|\|H_*^{-1}\mathbf{s}_k - \mathbf{y}_k\|\|M^{-1}\|}{\|M^{-1}\|\|M\mathbf{y}_k\|} \leq \frac{\|M\|\|M^{-1}\|}{\epsilon_2} l\sigma_{k,k+1}. \quad (4.7)$$

Moreover, as  $\alpha_k \leq \mu l\epsilon_1 < \frac{1}{2}$ , we have  $\omega_{k,M}^2 > \frac{1}{2}$  and therefore

$$\frac{1}{\omega_{k,M}^2} = 1 + \frac{1 - \omega_{k,M}^2}{\omega_{k,M}^2} \leq 1 + 2\alpha_k^2 \leq 1 + \alpha_k.$$

Thus (4.3) implies (B) for  $j = k$ . Finally the inequality (B) for  $j = 0, 1, \dots, k$  yields

$$\begin{aligned} \|H_{k+1} - H_*\|_{M^{-1}} - \|H_0 - H_*\|_{M^{-1}} &\leq \sum_{j=0}^k \|\mathcal{L}_{H_j} - H_j\|_{M^{-1}} \\ &\quad + \gamma_1 \epsilon_1 \sum_{j=0}^k r^j \|\mathcal{L}_{H_j} - H_j\|_{M^{-1}} \\ &\quad + \epsilon_1 [\gamma_1(\delta + \Omega) + \gamma_2] \sum_{j=0}^k r^j. \end{aligned}$$

We have shown that 1.-4. hold for  $j = k$ , provided that:

$$\epsilon_1 \leq \epsilon_3, \quad \rho\eta(\delta + \Omega) + (\eta(\delta + \Omega) + 2\mu)l\epsilon_1 \leq r, \quad \text{and}$$

$$\sum_{j=0}^{+\infty} \|\mathcal{L}_{H_j} - H_j\|_{M^{-1}} + \epsilon_1 \left( \gamma_1 \sum_{j=0}^{+\infty} r^j \|\mathcal{L}_{H_j} - H_j\|_{M^{-1}} + \frac{\gamma_1(\delta + \Omega) + \gamma_2}{1 - r} \right) \leq \Omega.$$

So the desired result follows as the latter conditions are simultaneously verified (for sufficiently small  $\epsilon_1$ ,  $\delta$ ) iff (4.4) holds.  $\square$

The prescription (4.4) can be verified only if the  $H_j$ 's converge to matrices of  $\mathcal{L}$  and this fact is consistent with the inequality  $\|H_{j+1} - H_*\|_{M^{-1}} < \delta + \Omega$  only if  $\|\mathcal{L}_{H_*} - \mathcal{L}_{H_*}\|$  is sufficiently small. Then, one could guess that the assertion (4.5) may not be verified for all  $r \in (0, 1)$  by the unmodified ( $\lambda_k = 1$ )  $\mathcal{IS}$   $\mathcal{LQN}$  methods, unless a restriction on  $\mathcal{L}$  is imposed. However, as it is shown in [14], no restriction on  $\mathcal{L}$  is needed to obtain the global convergence of  $\mathcal{NS}$   $\mathcal{LQN}$  methods. Thus, only the rate of convergence of  $\mathcal{LQN}$ -type methods may depend upon the choice of  $\mathcal{L}$ .

The next theorem 4.5 shows that, if  $\mathcal{L} = SDU_{\mathcal{L}}$  where  $U_{\mathcal{L}}$  is the unitary transform diagonalizing  $H_*$ , then the approximations  $\mathbf{x}_k$  generated by the corresponding unmodified  $\mathcal{IS}$  and  $\mathcal{INS}$   $\mathcal{LQN}$  methods, satisfy (4.5), i.e. the  $\mathbf{x}_k$ 's converge linearly to  $\mathbf{x}_*$ , for all  $r \in (0, 1)$ . Furthermore, for the  $\mathcal{INS}$  method, the  $\mathbf{x}_k$ 's converge superlinearly to  $\mathbf{x}_*$ . These results are obtained by using the following property of  $\mathcal{L}_H$ :

**Lemma 4.4.** Let  $\mathcal{L} = SDU_{\mathcal{L}}$  and let  $X \in \mathcal{L}$ . Set  $\|H\|_X = \|XHX\|_F$  where  $\|\cdot\|_F$  is the Frobenius norm. Then  $\|\mathcal{L}_H\|_X \leq \|H\|_X$ ,  $H \in \mathbb{C}^{n \times n}$ .

PROOF. The identity  $X\mathcal{L}_HX = \mathcal{L}_{XHX}$  [13] and the inequality  $\|\mathcal{L}_H\|_F \leq \|H\|_F$  [8] imply

$$\|\mathcal{L}_H\|_X = \|X\mathcal{L}_HX\|_F = \|\mathcal{L}_{XHX}\|_F \leq \|XHX\|_F = \|H\|_X. \quad \square$$

The assumption  $H_* = \mathcal{L}_{H_*}$  allows to choose  $M$  (such that  $M^2 = H_*$ ) in  $\mathcal{L}$ :  $M = U_{\mathcal{L}} \text{diag}(\sqrt{[U_{\mathcal{L}}^* H_* U_{\mathcal{L}}]_{jj}}) U_{\mathcal{L}}^*$ . So, by Proposition 2.1 and by Lemma 4.4,

$$\|\mathcal{L}_{H_k} - H_*\|_{M^{-1}} = \|\mathcal{L}_{H_k} - \mathcal{L}_{H_*}\|_{M^{-1}} = \|\mathcal{L}_{H_k - H_*}\|_{M^{-1}} \leq \|H_k - H_*\|_{M^{-1}}.$$

Thus one can replace  $\mathcal{L}_{H_k}$  by  $H_k$  on the right hand side of the inequality (4.3), thereby obtaining the inequalities (A),(B),(C). Consequently the result  $\|H_{j+1} - H_*\|_{M^{-1}} < \delta + \Omega$  holds without any assumption on the series  $\sum_{k=0}^{+\infty} \|\mathcal{L}_{H_k} - H_k\|$ .

In order to prove that  $\mathcal{INS}$  has a superlinear rate of convergence, one needs only to state the following inequality

$$\|H_{j+1} - H_*\|_{M^{-1}} \leq \left[ 1 - \frac{\alpha}{2} \theta_j^2 + \tilde{\gamma}_1 \sigma_{j,j+1} \right] \|\mathcal{L}_{H_j} - H_*\|_{M^{-1}} + \tilde{\gamma}_2 \sigma_{j,j+1}, \quad (\text{B}')$$

which is stronger than (B).

**Theorem 4.5.** Let  $\mathcal{L} = SDU_{\mathcal{L}}$  and  $H_* = \mathcal{L}_{H_*}$ . Let  $r$ ,  $0 < r < 1$ , be fixed. For any given  $\Omega$ ,  $0 < \Omega < \frac{r}{\eta\rho}$ , there exist  $\epsilon_1 = \epsilon_1(r)$ ,  $\delta = \delta(r)$  such that, if:

- $\|\mathbf{x}_0 - \mathbf{x}_*\| < \epsilon_1$ ,
- $H_0$  is pd and  $\|H_0 - H_*\|_{M^{-1}} < \delta$ ,
- $\mathbf{x}_{j+1} = \mathbf{x}_j - H_j \mathbf{g}_j$  ( $\mathcal{IS}$ ) and  $H_{j+1} = \Psi(\mathcal{L}_{H_j}, \mathbf{s}_j, \mathbf{y}_j)$ ,

then, for  $j = 0, 1, \dots$ , we have

1.  $\|H_j\| < \eta(\delta + \Omega) + \mu$  and  $\|H_j^{-1}\| < \frac{\rho}{1-r}$ ;
2. If  $\mathbf{x}_j \neq \mathbf{x}_*$ , then

$$\|\mathbf{x}_{j+1} - \mathbf{x}_*\| < r \|\mathbf{x}_j - \mathbf{x}_*\| < \epsilon_1,$$

otherwise the algorithm stops at the step  $j$ ;

3.  $\mathbf{y}_j^T \mathbf{s}_j > 0$  and then  $H_{j+1}$  is a well defined pd matrix;

4. (A), (B) and (C) hold and then  $\|H_{j+1} - H_*\|_{M^{-1}} < \delta + \Omega$ .

If  $H_0$  is such that  $\|\mathcal{L}_{H_0} - H_*\|_{M^{-1}} < \delta$  and  $\mathbf{x}_{j+1} = \mathbf{x}_j - \mathcal{L}_{H_j} \mathbf{g}_j$  ( $\mathcal{INS}$ ), then 1.-4. hold with  $\mathcal{L}_{H_j}$  in place of  $H_j$  in 1. and  $\mathcal{L}_{H_{j+1}}$  in place of  $H_{j+1}$  in 4..

Thus the BDM convergence result (4.1) holds for both the  $\mathcal{IS}$  and the  $\mathcal{INS}$  LQN sequences  $\{\mathbf{x}_k\}$  provided that  $H_* = \mathcal{L}_{H_*}$ . Moreover, in the  $\mathcal{INS}$  case

$$\lim_{k \rightarrow \infty} \frac{\|\mathbf{x}_{k+1} - \mathbf{x}_*\|}{\|\mathbf{x}_k - \mathbf{x}_*\|} = 0,$$

i.e.  $\{\mathbf{x}_k\}$  converges superlinearly to  $\mathbf{x}_*$ .

PROOF. One can easily show that, in both  $\mathcal{IS}$  and  $\mathcal{INS}$  cases, the thesis holds if

$$\begin{aligned} \epsilon_1 &\leq \epsilon_3, \quad \rho\eta(\delta + \Omega) + (\eta(\delta + \Omega) + 2\mu)l\epsilon_1 \leq r, \\ [\gamma_1(\delta + \Omega) + \gamma_2] \frac{\epsilon_1}{1-r} &\leq \Omega \end{aligned}$$

(proceed essentially as in Theorem 4.3). For sufficiently small  $\epsilon_1, \delta$  these conditions are clearly satisfied. Hence, at least the BDM convergence result is guaranteed. Notice the different use of the result

$$H_j \text{ pd} \Rightarrow \mathcal{L}_{H_j} \text{ pd} \quad (4.8)$$

in the proof of point 3.. In the  $\mathcal{IS}$  case one exploits (4.8) only to deduce from the inequality  $\mathbf{y}_j^T \mathbf{s}_j > 0$  that  $H_{j+1}$  is  $pd$  (see (2.8)). In the  $\mathcal{INS}$  case the same inequality  $\mathbf{y}_j^T \mathbf{s}_j > 0$  is derived by (4.8).

Consider now the  $\mathcal{INS}$ . Let us prove that the rate of convergence of  $\mathbf{x}_k$  to  $\mathbf{x}_*$  is superlinear if  $\epsilon_1 = \epsilon_1(r)$  is also such that  $\frac{\gamma_2 \epsilon_1}{4} \leq \frac{1}{3}$  and  $\mathbf{x}_j \neq \mathbf{x}_*$ ,  $\forall j$ . Thus  $0 < \|\mathbf{x}_{j+1} - \mathbf{x}_*\| < r \|\mathbf{x}_j - \mathbf{x}_*\|$ ,  $\mathbf{s}_j \neq \mathbf{0}$ ,  $\mathbf{y}_j \neq \mathbf{0}$ ,  $j = 0, 1, \dots$ , and, by (4.7),

$$\|M^{-1} \mathbf{s}_j - M \mathbf{y}_j\| \leq \frac{1}{3} \|M \mathbf{y}_j\|, \quad \forall j. \quad (4.9)$$

Assume firstly that a subsequence of  $\{\|\mathcal{L}_{H_j} - H_*\|_{M^{-1}}\}$  converges to zero. Let  $\bar{r}$  be any fixed real number in  $(0, 1)$  and let  $\epsilon_1(\bar{r})$ ,  $\delta(\bar{r})$  be the corresponding positive numbers defined in the statement of this theorem. Since there exists an index  $\bar{m} = \bar{m}(\bar{r})$  such that  $\|\mathbf{x}_{\bar{m}} - \mathbf{x}_*\| < \epsilon_1(\bar{r})$  and  $\|\mathcal{L}_{H_{\bar{m}}} - H_*\|_{M^{-1}} < \delta(\bar{r})$ , by assertion 2.

$$\|\mathbf{x}_{j+1} - \mathbf{x}_*\| < \bar{r} \|\mathbf{x}_j - \mathbf{x}_*\|, \quad \forall j \geq \bar{m}.$$

Consider now the case  $\|\mathcal{L}_{H_j} - H_*\|_{M^{-1}} \geq \text{cost} > 0, \forall j$ . (4.9) allows us to apply Lemma 4.2 in [7] (or Lemma 3.1 in [11]), thereby obtaining

$$\begin{aligned} \|H_{j+1} - H_*\|_{M^{-1}} &\leq \left[ \sqrt{1 - \alpha \theta_j^2} + \alpha_1 \frac{\|M^{-1} \mathbf{s}_j - M \mathbf{y}_j\|}{\|M \mathbf{y}_j\|} \right] \|\mathcal{L}_{H_j} - H_*\|_{M^{-1}} \\ &\quad + \alpha_2 \frac{\|\mathbf{s}_j - H_* \mathbf{y}_j\|}{\|M \mathbf{y}_j\|} \end{aligned} \quad (4.10)$$

where  $\theta_j = \frac{\|M^{-1}(\mathcal{L}_{H_j} - H_*) \mathbf{y}_j\|}{\|\mathcal{L}_{H_j} - H_*\|_{M^{-1}} \|M \mathbf{y}_j\|}$  and  $\alpha = \frac{3}{8}$ ,  $\alpha_1 = \frac{15}{4}$ ,  $\alpha_2 = 2(1 + 2\sqrt{n})\|M^{-1}\|$ . Thus

$$\|H_{j+1} - H_*\|_{M^{-1}} \leq \left[ \sqrt{1 - \alpha \theta_j^2} + \tilde{\gamma}_1 \sigma_{j,j+1} \right] \|\mathcal{L}_{H_j} - H_*\|_{M^{-1}} + \tilde{\gamma}_2 \sigma_{j,j+1}$$

where  $\tilde{\gamma}_1 = \alpha_1 \frac{\gamma_2}{4}$  and  $\tilde{\gamma}_2 = \alpha_2 \frac{\gamma_2}{4} \|M\|$ , and (B') is obtained for all  $j$ . For  $j = 0, 1, \dots, k$  the relations  $\|\mathcal{L}_{H_{j+1}} - H_*\|_{M^{-1}} \leq \|H_{j+1} - H_*\|_{M^{-1}}$  and (B') yield the inequality

$$\frac{\alpha}{2} \sum_{j=0}^k \theta_j^2 \|\mathcal{L}_{H_j} - H_*\|_{M^{-1}} \leq \|\mathcal{L}_{H_0} - H_*\|_{M^{-1}} + \epsilon_1(\tilde{\gamma}_1(\delta + \Omega) + \tilde{\gamma}_2) \frac{1}{1-r},$$

so the series  $\sum_{j=0}^{+\infty} \theta_j^2 \|\mathcal{L}_{H_j} - H_*\|_{M^{-1}}$  is convergent. Since  $\|\mathcal{L}_{H_j} - H_*\|_{M^{-1}} < \delta + \Omega$ , it follows

$$\frac{\|(\mathcal{L}_{H_j} - H_*)\mathbf{y}_j\|}{\|\mathbf{y}_j\|} \leq \|M\|^2 \frac{\|M^{-1}(\mathcal{L}_{H_j} - H_*)\mathbf{y}_j\|}{\|M\mathbf{y}_j\|} \rightarrow 0. \quad (4.11)$$

On the other hand, by assertion 1. and Lemma 4.1, we have

$$\begin{aligned} \mathcal{L}_{H_j}\mathbf{y}_j &= \mathcal{L}_{H_j}\mathbf{g}_{j+1} + \mathbf{x}_{j+1} - \mathbf{x}_j \Rightarrow \\ \mathbf{g}_{j+1} &= \mathcal{L}_{H_j}^{-1}[(\mathcal{L}_{H_j} - H_*)\mathbf{y}_j + H_*(\mathbf{y}_j - H_*^{-1}\mathbf{s}_j)] \Rightarrow \\ \|\mathbf{g}_{j+1}\| &\leq \frac{\rho}{1-r} (\|(\mathcal{L}_{H_j} - H_*)\mathbf{y}_j\| + \|H_*\|l\|\mathbf{x}_j - \mathbf{x}_*\|\|\mathbf{s}_j\|). \end{aligned}$$

Thus

$$\frac{\|\mathbf{g}_{j+1} - \mathbf{g}(\mathbf{x}_*)\|}{\|\mathbf{s}_j\|} = \frac{\|\mathbf{g}_{j+1}\|}{\|\mathbf{s}_j\|} \leq \frac{\rho}{1-r} \left( \frac{\|(\mathcal{L}_{H_j} - H_*)\mathbf{y}_j\|}{\epsilon_2 \|\mathbf{y}_j\|} + \|H_*\|l\|\mathbf{x}_j - \mathbf{x}_*\| \right)$$

and consequently

$$\frac{\epsilon_2 \|\mathbf{x}_{j+1} - \mathbf{x}_*\|}{2 \|\mathbf{x}_j - \mathbf{x}_*\|} \leq \frac{\rho}{1-r} \left( \frac{\|(\mathcal{L}_{H_j} - H_*)\mathbf{y}_j\|}{\epsilon_2 \|\mathbf{y}_j\|} + \|H_*\|l\|\mathbf{x}_j - \mathbf{x}_*\| \right).$$

By (4.11) and the latter inequality, we finally obtain the thesis.

Theorem 4.5 states that the  $\mathcal{INS-LQN}$  algorithm,  $\mathcal{L} = SDU_{\mathcal{L}}$ , computes the solution  $\mathbf{x}_*$  with the same number of steps  $\bar{k}$  required by  $BFGS$ , provided that  $U_{\mathcal{L}}$  diagonalizes  $H_*$ . Moreover, by Theorem 3.2, if  $U_{\mathcal{L}}$  defines a fast transform, then the same algorithm can be implemented with only  $O(n \log n)$  arithmetic operations per step and  $O(n)$  memory allocations. So, under these two assumptions on  $U_{\mathcal{L}}$ , we can immediately conclude that  $\mathcal{INS-LQN}$  outperforms  $BFGS$  whose complexity is  $O(n^2)$ .

## 5. Final remarks

The experimental data show that the secant  $LQN$  methods have a rate of convergence more than linear, independently from the fact if  $\|\nabla^2 f(\mathbf{x}_*) - \mathcal{L}_{\nabla^2 f(\mathbf{x}_*)}\|$  is small or great. They are, in fact, competitive with the best known methods for large scale optimization problems [4]. Moreover, if  $\mathcal{L}$  includes the matrix  $\nabla^2 f(\mathbf{x}_*)$ , i.e.  $\nabla^2 f(\mathbf{x}_*) = \mathcal{L}_{\nabla^2 f(\mathbf{x}_*)}$ , then, by Theorem 4.5, the  $\mathcal{INS-LQN}$  rate of convergence is superlinear. Such choice of  $\mathcal{L}$  is not necessary in order to obtain the global convergence of  $\mathcal{NS-LQN}$  [14]. However, for such  $\mathcal{L}$ , the  $\mathcal{ILQN}$  methods seem to be even superior to  $BFGS$  (see Table 2 in Section 3). So, the rate of convergence of  $LQN$  and  $\mathcal{ILQN}$  may, in fact, depend upon  $\mathcal{L}$ .

The previous results let one guess that a suitable choice of  $\mathcal{L}$  can improve the performance, in terms of number of iterations, of  $LQN$ -type methods. The same Theorem 4.3, where the Broyden-Dennis-Moré convergence result is extended to  $\mathcal{IS-LQN}$  methods under the condition (4.4), suggests that an adaptive choice of  $\mathcal{L}$  during the minimization process is perhaps the best way to improve the  $\mathcal{ILQN}$  and  $LQN$  rates of convergence. As a matter of fact, one observes that the proof and the statement of *Theorem 4.3 hold unchanged if the fixed space  $\mathcal{L}$  is replaced with a space  $\mathcal{L}^{(k)}$* , which may vary with  $k$ . So we obtain, instead of (4.4), a weaker BDM convergence condition,

$$\sum_{j=0}^{+\infty} \|(\mathcal{L}^{(j)})_{H_j} - H_j\|_{M^{-1}} < \Omega < \frac{r}{\eta\rho}, \quad (5.1)$$

and we can try to define the spaces  $\mathcal{L}^{(k)}$  so that (5.1) is not far from being satisfied. In the context of  $SDU_{\mathcal{L}}$  spaces, where  $(\mathcal{L}^{(j)})_{H_j}$  are in fact *pd* matrices, a possible choice of  $\mathcal{L}^{(k+1)}$  could be  $\mathcal{L}^{(k+1)} = SDU_{\mathcal{L}^{(k+1)}}$  with  $U_{\mathcal{L}^{(k+1)}}$  derived from  $U_{\mathcal{L}^{(k)}}$  by a suitable updating procedure, conceived so that  $\|(\mathcal{L}^{(k+1)})_{H_{k+1}} - H_{k+1}\|$  is small and the space  $\mathcal{L}^{(k+1)}$  is (almost) structured as  $\mathcal{L}^{(k)}$ . This procedure may finally result in a definitive  $\mathcal{IS-L^*QN}$  method, optimal for  $\mathbf{x}_*$ . The possibility of defining a procedure *updating fast transforms*  $U_{\mathcal{L}^{(k)}}$  is supported by the existence of a formula updating the Cholesky factors of the *BFGS* Hessian approximations [12] and by the existence of continuous sets  $\mathcal{F}$  of discrete fast transforms. Concerning the latter point, the same splitting formulas on which are based the fast algorithms computing the classical Fourier and the more recent [3] Hartley-type transforms can be exploited to generate  $\mathcal{F}$ , through the following steps ( $n = 2^s$ ):

1. Choose  $A_i^{(2)} \in \mathbb{C}^{2 \times 2}$ ,  $i = 0, \dots, 2^{s-1} - 1$ , unitary (not necessarily equal);
2. For  $l = 1, \dots, s-1$  and  
For  $k = 0, \dots, 2^{s-l-1} - 1$  set

$$A_k^{(2^{l+1})} = \frac{1}{\sqrt{2}} \begin{bmatrix} R_{0,0,k}^{(2^l)} & R_{0,1,k}^{(2^l)} \\ R_{1,0,k}^{(2^l)} & R_{1,1,k}^{(2^l)} \end{bmatrix} \begin{bmatrix} A_{2k}^{(2^l)} & 0 \\ 0 & A_{2k+1}^{(2^l)} \end{bmatrix} Q^{(2^{l+1})}$$

with  $Q^{(2^{l+1})}$  equal to the even-odd permutation matrix and  $R_{i,j,k}$  such that

- $A_k^{(2^{l+1})}$  is unitary,
  - each row of  $R_{i,j,k}$  has at most  $c$  nonzero entries;
3. Define  $U_{\mathcal{L}} = A_0^{(2^s)} \in \mathbb{C}^{n \times n}$ .

The resulting unitary matrix  $U_{\mathcal{L}}$  and the corresponding space  $\mathcal{L} = SDU_{\mathcal{L}}$  are not necessarily well structured, like the Fourier matrix  $F$  and the space of circulants  $\mathcal{C}$ . However,  $U_{\mathcal{L}}$  is defined in terms of at most  $O(n \log n)$  parameters and can be multiplied by a vector in  $O(n \log n)$  arithmetic operations. This is sufficient to state that the corresponding  $\mathcal{IS-(SDU_{\mathcal{L}})QN}$  method can be competitive with  $\mathcal{IS-(SDU_{\mathcal{C}})QN}$  in solving large scale unconstrained minimization problems.

*Acknowledgement. I'm happy to thank professor Eugene Tyrtyshnikov and the organizing committee of the conference. I'll never forget those wonderful days in USA.*

## References

- [1] R. Battiti, First- and second-order methods for learning: between steepest descent and Newton's method, *Neural Computation* **4** (1992) 141–166.
- [2] D. Bini, P. Favati, On a matrix algebra related to the discrete Hartley transform. *SIAM J. Matrix Anal. Appl.* **14** (1993) 500–507.
- [3] A. Bortoletti, C. Di Fiore, On a set of matrix algebras related to discrete Hartley-type transforms. *Linear Algebra Appl.*, to appear.
- [4] A. Bortoletti, C. Di Fiore, S. Fanelli, P. Zellini. A new class of quasi-Newtonian methods for optimal learning in *MLP*-networks, *IEEE Transactions on Neural Networks*, to appear.
- [5] E. Bozzo, C. Di Fiore, On the use of certain matrix algebras associated with discrete trigonometric transforms in matrix displacement decomposition, *SIAM J. Matrix Anal. Appl.* **16** (1995) 312–326.
- [6] R. N. Bracewell, The fast Hartley transform. *Proc. IEEE* **72** (1984) 1010–1018.

- [7] C. G. Broyden, J. E. Dennis, Jr., Jorge J. Moré, On the local and superlinear convergence of quasi-Newton methods, *J. Inst. Maths Applics.* **12** (1973) 223–245.
- [8] R. Chan, X. Jin, M. Yeung, The circulant operator in the Banach algebra of matrices, *Linear Algebra Appl.* **149** (1991) 41–53.
- [9] T. F. Chan, An optimal circulant preconditioner for Toeplitz systems, *SIAM J. Sci. Stat. Comput.* **9** (1988) 766–771.
- [10] J. E. Dennis, Jr., Jorge J. Moré, Quasi-Newton methods, motivation and theory, *SIAM Review* **19** (1977) 46–89.
- [11] J. E. Dennis, Jr., Jorge J. Moré, A characterization of superlinear convergence and its application to quasi-Newton methods, *Math. Comp.* **28** (1974) 549–560.
- [12] J. E. Dennis, Jr., R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations* (Prentice-Hall, Englewood Cliffs, New Jersey, 1983).
- [13] F. Di Benedetto, S. Serra Capizzano, Optimal and superoptimal matrix algebra operators, Dept. of Mathematics - Univ. of Genova, Italy, TR nr. **360** (1997).
- [14] C. Di Fiore, S. Fanelli, F. Lepore, P. Zellini, Matrix algebras in quasi-Newton methods for unconstrained minimization, *Numerische Mathematik*, to appear.
- [15] C. Di Fiore, S. Fanelli, P. Zellini, Matrix algebras in quasi-newtonian algorithms for optimal learning in Multi-Layer Perceptrons, Proc. of the ICONIP/ANZIIS/ANNES '99 (Dunedin, New Zealand, Nov. 1999), N.Kasabov, K.Ko eds. (1999) 27–32.
- [16] C. Di Fiore, F. Lepore, P. Zellini, Hartley-type algebras in displacement, preconditioning and optimization strategies, *Linear Algebra Appl.*, to appear.
- [17] C. Di Fiore, P. Zellini, Matrix algebras in optimal preconditioning, *Linear Algebra Appl.* **335** (2001) 1–54.
- [18] S. Fanelli, P. Paparo, M. Protasi, Improving performances of Battiti-Shanno's quasi-newtonian algorithms for learning in feed-forward neural networks, Proc. of the 2nd Australian and New Zealand Conference on Intelligent Information Systems (Brisbane, Australia, Nov.-Dec. 1994), (1994) 115–119.
- [19] N. C. Hu, H. I. Chang, O. K. Ersoy, Generalized discrete Hartley transforms, *IEEE Trans. Signal Process.* **40** (12) (1992) 2931–2940.
- [20] 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) 191–208.
- [21] J. Nocedal, S. J. Wright, *Numerical Optimization* (Springer, New York, 1999).
- [22] J. M. Ortega, W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables* (Academic Press, New York, 1970).
- [23] D. Potts, G. Steidl, Optimal trigonometric preconditioners for nonsymmetric Toeplitz systems, *Linear Algebra Appl.* **281** (1998) 265–292.
- [24] M. J. D. Powell, On the convergence of the variable metric algorithm, *J. Inst. Maths Applics.* **7** (1971) 21–36.
- [25] M. J. D. Powell, Some global convergence properties of a variable metric algorithm for minimization without exact line searches, *Nonlinear Programming*, SIAM-AMS Proc. (New York, March 1975), R. W. Cottle, C. E. Lemke eds., Providence, **9** (1976) 53–72.
- [26] D. F. Shanno, Conjugate gradient methods with inexact searches, *Math. Oper. Res.* **3** (1978) 244–256.
- [27] E. E. Tyrtyshnikov, Optimal and superoptimal circulant preconditioners, *SIAM J. Matrix Anal. Appl.* **13** (1992) 459–473.
- [28] E. E. Tyrtyshnikov, Matrix approximations: theory and applications in numerical analysis, Lectures held in the Dept. of Mathematics and Computer Science of the Univ. of Udine, October, 2000.
- [29] P. R. Uniyal, Transforming real-valued sequences: fast Fourier versus fast Hartley transform algorithms, *IEEE Trans. Signal Process.* **42** (11) (1994) 3249–3254.
- [30] Z. Wang, Fast algorithms for the discrete  $W$  transform and for the discrete Fourier transform, *IEEE Trans. Acoust. Speech Signal Processing* **ASSP-32** (1984) 803–816.

*This page intentionally left blank*

## Computation of minimal state space realizations in Jacobson normal form

Naoharu Ito, Wiland Schmale, and Harald K. Wimmer

**ABSTRACT.** Given a strictly proper rational transfer matrix over an arbitrary field a minimal realization  $(F, G, H)$  is constructed with  $F$  being in Jacobson normal form. The procedure extends Kalman's invariant factor approach. It is particularly well suited for finite fields where factorization of polynomials can be done effectively. The computation relies on the partial fraction decomposition and the Smith-McMillan form.

### 1. Introduction

Let

$$(1.1) \quad (F \in K^{\ell \times \ell}, G \in K^{q \times \ell}, H \in K^{\ell \times t})$$

be a triple of matrices over a field  $K$  and let

$$(1.2) \quad x(k+1) = Fx(k) + Gu(k), \quad y(k) = Hx(k),$$

be the corresponding discrete-time system with inputs  $u(k) \in K^q$ , states  $x(k) \in K^\ell$  and outputs  $y(k) \in K^t$ . The system (1.2) with initial state  $x(0) = 0$  and input sequence  $(u(k))_{k \in \mathbb{N}_0}$  generates an output sequence  $(y(k))_{k \in \mathbb{N}}$  with

$$y(k) = \sum_{i=0}^{k-1} HF^{k-1-i}Gu_i.$$

Let

$$u(s) = \sum_{k=0}^{\infty} u(k)s^{-k} \text{ and } y(s) = \sum_{k=1}^{\infty} y(k)s^{-k}$$

be the associated formal power series. Then the input-output map defined by (1.2) is

$$y(s) = W(s)u(s)$$

---

1991 *Mathematics Subject Classification.* Primary 93B20, 93B15, 15A23; Secondary 15A21 .

*Key words and phrases.* Minimal realizations, Jacobson normal form, Smith-McMillan form,  $p$ -adic expansions.

The first author was supported by the Japanese Society for the Promotion of Sciences.

with transfer matrix

$$W(s) = \sum_{k=0}^{\infty} HF^k G = H(sI - F)^{-1} G.$$

Note that  $W \in K^{q \times t}(s)$  is a matrix of *strictly proper* rational functions, i.e. in each nonzero entry of  $W$  the degree of the denominator polynomial is greater than the degree of the numerator. Now consider an inverse problem. Let  $W \in K^{q \times t}(s)$  be a given strictly proper rational transfer matrix. Can  $W$  be “realized” by a linear system of the form (1.2), i.e. can  $W$  be represented as

$$(1.3) \quad W(s) = G(sI - F)^{-1} H.$$

A factorization (1.3) with  $F \in K^{\ell \times \ell}$  is called a *state space realization* of dimension  $\ell$  of  $W$ . The realization (1.3) is *minimal* if the dimension  $\ell$  is the lowest possible. We refer to [dS] for a survey of results on minimal state space realizations and an extensive bibliography. Basic facts and concepts of linear systems theory can be found in [Kt], [KF] or [Ro].

It is well known that a strictly proper rational matrix  $W$  admits a state space realization (1.3). According to Kalman’s state space isomorphism theorem (see e.g. [Fu, p.282]) the matrix  $F$  in (1.3) is uniquely determined up to similarity if the realization is minimal. Hence, if the field  $K$  is algebraically closed there exists a minimal realization (1.3) such that the matrix  $F$  is in Jordan normal form. Kalman [Ka] constructed such a realization by transforming the principal parts of a complex transfer matrix  $W$  into Smith-McMillan form and using Laurent expansions at poles of  $W$ . If the field  $K$  is different from  $\mathbb{C}$  (or  $\mathbb{R}$ ) then the appropriate normal form which displays the elementary divisor structure of  $F$  in (1.3) is the Jacobson form. It is the objective of this paper to extend Kalman’s approach to strictly proper rational matrices over an arbitrary field  $K$  and to compute a realization (1.3) where  $F$  appears in Jacobson normal form. We shall proceed as follows. In Section 2 we give a short account of the Jacobson normal form. In Section 3 we prepare the mathematical tools for the realization procedure in Section 4. An example is given in Section 5.

The motivation for our study comes from applications of systems over finite fields such as linear sequential circuits [Gi]. As convolutional codes can be interpreted as linear sequential circuits (see e.g. [MS], [Fo], [Rt], [RS]) our algorithm could become useful for constructions of codes along the lines of [RY].

In [KF, p.288] we find the following comment on Kalman’s algorithm [Ka]. “The solution of the realization problem via the invariant factor algorithm provides complete information concerning the structure of the canonical realization. Unfortunately, this method can be applied only at the cost of very tedious and complex computations.” We remark that in the case of a finite field  $K$  prime factorization of polynomials and computation of the Smith-McMillan form is performed accurately and efficiently by today’s computer algebra software. Hence in that case realizations with  $F$  in Jacobson normal form do not encounter the computational difficulties indicated in [KF].

## 2. The Jacobson normal form

Let us briefly recall how the Jacobson normal form generalizes the concept of Jordan normal form. Let  $p \in K[s]$  be a monic polynomial,

$$p(s) = s^n + a_{n-1}s^{n-1} + \cdots + a_0.$$

and let

$$(2.1) \quad C = C(p) = \begin{pmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ -a_0 & \dots & -a_{n-1} & \end{pmatrix}$$

be the companion matrix associated with  $p$ . Define

$$(2.2) \quad V = \begin{pmatrix} 0 & 0 & \dots & 0 \\ \vdots & & & \vdots \\ 1 & 0 & \dots & 0 \end{pmatrix}_{n \times n} = e_n e_1^T,$$

where  $e_1 = (1, 0, \dots, 0)^T$  and  $e_n = (0, \dots, 0, 1)^T$  are unit vectors of  $K^n$ . We call

$$(2.3) \quad J = J(p^k) = \begin{pmatrix} C & V & & 0 \\ 0 & C & & \vdots \\ \vdots & & \ddots & \vdots \\ \vdots & & & C & V \\ 0 & & 0 & C \end{pmatrix}_{nk \times nk}$$

a *Jacobson block* corresponding to  $p^k$ . If  $p = s - \lambda$ , then  $C(p) = (\lambda)_{1 \times 1}$ . Hence the Jordan block

$$J[(s - \lambda)^k] = \begin{pmatrix} \lambda & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & \lambda \end{pmatrix}_{k \times k}$$

is a special case of (2.3). The fact that the  $(i, i+1)$ -entries of  $J(p^k)$  are equal to 1 implies that  $J(p^k)$  is nonderogatory and that the Smith form of  $sI - J(p^k)$  is  $\text{diag}(1, \dots, 1, p^k)$ . Hence, if  $A \in K^{\ell \times \ell}$  has  $p^k$  as its only elementary divisor, then  $A$  is similar (over  $K$ ) to  $J(p^k)$ . The following, more general result was proved by Krull in his Ph.D. thesis [Kr]. It can be found in [Ay, p. 295], [Ja, p. 72-73], [Co, p. 300] or [AW, p.252].

**THEOREM 2.1.** *Let  $p_1, \dots, p_m$  be the distinct irreducible factors of the characteristic polynomial of a matrix  $A \in K^{\ell \times \ell}$  and let*

$$(2.4) \quad p_1^{k_{11}}, \dots, p_1^{k_{1\tau_1}}, \dots, p_m^{k_{m1}}, \dots, p_m^{k_{m\tau_m}},$$

$$k_{11} \leq \cdots \leq k_{1\tau_1}, \dots, k_{m1} \leq \cdots \leq k_{m\tau_m},$$

*be the corresponding elementary divisors of  $sI - A$ . Then  $A$  is similar to*

$$(2.5) \quad \text{diag}\left(J(p_1^{k_{11}}), \dots, J(p_m^{k_{m\tau_m}})\right).$$

*The matrix (2.5) is unique up to the order of the polynomials  $p_1, \dots, p_m$ .*

It is common to call the matrix (2.5) the *Jacobson normal form* of  $A$ . Some authors, e.g. [AW], call (2.5) a *generalized Jordan canonical form*. Computational aspects of the Jacobson form have been considered in e.g. in [DJ] and [KM]. In [DJ] the resolvent  $(sI - A)^{-1}$  is used to compute the Jacobson normal form of  $A$ , the approach in [KM] is based on the rational canonical form of  $A$ .

### 3. Notation and background

Let  $K_{sp}(s)$  denote the  $K$ -vector space of strictly proper rational functions over  $K$ . Then each  $f \in K(s)$  can be decomposed uniquely as

$$f = w + y$$

such that  $w \in K_{sp}(s)$  and  $y \in K[s]$ . If we set  $\pi_- f = w$ , then  $\pi_-$  is the projection of  $K(s)$  onto  $K_{sp}(s)$ . In a natural way the definitions of  $\pi_-$  and  $K_{sp}(s)$  extend element-wise to vectors and matrices of rational functions. The gcd of two polynomials  $a, b$  will be denoted by  $(a, b)$ . For a nonzero polynomial vector  $h = (h_1, \dots, h_r)^T \in K^r[s]$  we define

$$\deg h = \max\{\deg h_i, h_i \neq 0, i = 1, \dots, r\}.$$

We set  $\deg h = -\infty$  if  $h = 0$ . Let  $R$  be a matrix of rational functions. Then  $R = PA^{-1}S$  is called a *coprime factorization* of  $R$  if  $P, S, A$  are polynomial matrices such that the pairs  $(A, P)$  and  $(A, S)$  are right and left coprime, respectively.

The building blocks of our realization will be obtained from the following lemma. In addition to the companion matrix  $C$  associated with  $p(s) = s^n + a_{n-1}s^{n-1} + \dots + a_0$  we shall need the matrix

$$(3.1) \quad M = M(p) = \begin{pmatrix} a_1 & a_2 & \dots & \dots & a_{n-1} & 1 \\ a_2 & a_3 & \dots & \dots & 1 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ a_{n-1} & 1 & \ddots & \ddots & \ddots & 0 \\ 1 & 0 & \ddots & \ddots & \ddots & 0 \end{pmatrix}.$$

Note that  $M(p)$  satisfies  $M(p)C(p) = C^T(p)M(p)$ .

LEMMA 3.1. [IS] *Let  $W \in K^{q \times t}(s)$ ,  $W \neq 0$ , be of rank 1,*

$$(3.2) \quad W = h \frac{1}{p^k} g^T$$

*where  $h \in K^q[s]$ ,  $g \in K^t[s]$  are polynomial vectors. Let  $h$  have the  $p$ -adic expansion*

$$(3.3) \quad h = h_0 + h_1 p + \dots + h_{k-1} p^{k-1} + \dots$$

*where*

$$(3.4) \quad h_i \in K^q[s], \deg h_i < n = \deg p, i \geq 0.$$

*Define  $H_i \in K^{q \times n}$ ,  $i \geq 0$ , by*

$$(3.5) \quad h_i(s) = H_i \begin{pmatrix} 1 \\ s \\ \vdots \\ s^{n-1} \end{pmatrix}.$$

Given the expansion

$$(3.6) \quad g = g_0 + g_1 p + \cdots + g_{k-1} p^{k-1} + \dots$$

with

$$(3.7) \quad g_i \in K^t[s], \deg g_i < n, i \geq 0,$$

define matrices  $G_i \in K^{t \times n}$  by

$$(3.8) \quad g_i(s) = G_i M(p) \begin{pmatrix} 1 \\ s \\ \vdots \\ s^{n-1} \end{pmatrix}.$$

Set

$$H = (H_0, \dots, H_{k-1})$$

and

$$G = \begin{pmatrix} G_{k-1}^T \\ \vdots \\ G_0^T \end{pmatrix}.$$

Then

$$(3.9) \quad \pi_- W = H \left( sI - J(p^k) \right)^{-1} G.$$

The realization in (3.9) is minimal if (3.2) is a coprime factorization.

If  $W \in \mathbb{C}^{q \times t}(s)$  is a complex rational matrix with poles  $\lambda_1, \dots, \lambda_m$  then

$$W = \sum_{i=1}^m W_{s-\lambda_i}$$

where  $W_{s-\lambda_i}$  is the principal part of  $W$  at  $s = \lambda_i$ . In the case of an arbitrary field  $K$  the corresponding decomposition involves irreducible polynomials, possibly of degree greater than 1. The following fact is an immediate consequence of the partial fraction decomposition in  $K(s)$ .

LEMMA 3.2. *Let  $W \in K^{q \times t}(s)$  be strictly proper and let  $\mathcal{P}(W)$  be the set of monic irreducible divisors of the least common denominator of the entries of  $W$ . Then there is a unique decomposition*

$$(3.10) \quad W = \sum_{p \in \mathcal{P}(W)} W_p$$

such that  $W_p$  is strictly proper and  $\mathcal{P}(W_p) = \{p\}$ .

We call the component  $W_p$  the *principal part* of  $W$  at  $p$ . The next observation shows that a realization of  $W$  can be obtained from realizations of its principal parts.

LEMMA 3.3. [Ka] *Let  $W \in K^{q \times t}(s)$  be decomposed into principal parts such that*

$$(3.11) \quad W = \sum_{\mu=1}^m W_{p_\mu}$$

where  $\mathcal{P}(W) = \{p_1, \dots, p_m\}$  and  $\mathcal{P}(W_{p_\mu}) = \{p_\mu\}$ . If

$$(3.12) \quad H_\mu(sI - F_\mu)^{-1}G_\mu = W_{p_\mu}(s), \quad \mu = 1, \dots, m.$$

are minimal realizations and if we set

$$F = \text{diag}(F_1, \dots, F_m), \quad G = \begin{pmatrix} G_1 \\ \vdots \\ G_m \end{pmatrix}, \quad H = (H_1, \dots, H_m).$$

then

$$(3.13) \quad H(sI - F)^{-1}G = W(s)$$

is a minimal realization.

We call (3.13) the *direct sum* of the realizations (3.12). In the next lemma the Smith-McMillan form is used to reduce the realization problem to the case of rational matrices of rank 1.

**LEMMA 3.4. [Ka]** *Let  $W \in K^{q \times t}(s)$  be a strictly proper rational matrix of rank  $r$ . Assume  $\mathcal{P}(W) = \{p\}$ , i.e. the common denominator of the entries of  $W$  is a power of an irreducible polynomial  $p$ . Let*

$$(3.14) \quad \Sigma = \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix}$$

be the Smith-McMillan form of  $W$  with

$$(3.15) \quad D = \text{diag}\left(\frac{a_1}{p^{k_1}}, \dots, \frac{a_r}{p^{k_r}}\right), \quad k_1 \geq \dots \geq k_r \geq 0.$$

and

$$(3.16) \quad a_i \in K[s], \quad (p, a_i) = 1, \quad i = 1, \dots, r, \quad a_1 | \dots | a_r.$$

Then the dimension of a minimal realization of  $W$  is equal to

$$(3.17) \quad \sum_{i=1, k_i \geq 1}^r n k_i.$$

Let  $U = (u_1, \dots, u_q) \in K^{q \times q}[s]$  and  $V = (v_1, \dots, v_t) \in K^{t \times t}[s]$  be unimodular matrices such that

$$(3.18) \quad W = U\Sigma V^T.$$

Set

$$(3.19) \quad w_i = u_i \frac{a_i}{p^{k_i}} v_i^T, \quad i = 1, \dots, r.$$

Then

$$(3.20) \quad W = \sum_{i=1, k_i \geq 1}^r \pi_- w_i.$$

The polynomials  $p^{k_1}, \dots, p^{k_r}$  in the Smith-McMillan form (3.15) are the invariant factors of  $F$  if  $(F, G, H)$  is a minimal realization of  $W$  [Ka].

When  $w_i$  in (3.19) is not strictly proper then it expedites the computation if remainders mod  $p^k$  are used. E.g., in the realization of Section 4 we compute  $\tilde{a}_i \in K[s]$  such that

$$(3.21) \quad \pi_-(a_i p^{-k_i}) = \tilde{a}_i p^{-k_i}.$$

Then  $\pi_- w_i = \pi_-(u_i \tilde{a}_i) p^{-k_i} v_i^T$ .

#### 4. The algorithm

In the following procedure we refer to lemmas and definitions of Section 3. Let  $W \in K^{q \times t}(s)$  be a given strictly proper rational matrix.

**Step 1.** Decompose  $W$  as in (3.10) as the sum of its principal parts  $W_p$ .

**Step 2.** For each  $p \in \mathcal{P}(W)$  compute a minimal realization  $(F_p, G_p, H_p)$  of  $W_p$  in the following way.

**Step 2.1.** Set  $W = W_p$  in Lemma 3.4. Compute  $U, V, \Sigma$  such that (3.14) - (3.18) hold. Compute the rank 1 matrices  $w_i$ ,  $i = 1, \dots, m$ , given by (3.19).

**Step 2.2.** Set  $W = w_i$  in Lemma 3.1. Compute the realization (3.9) for  $i = 1, \dots, m$ .

**Step 2.3.** Compute a realization  $(F_p, G_p, H_p)$  of  $W = W_p$  as the direct sum of the realizations of  $w_i$ ,  $i = 1, \dots, m$ .

**Step 3.** Compute a minimal realization of  $W$  as the direct sum of the realizations  $(F_p, G_p, H_p)$ ,  $p \in \mathcal{P}(W)$ .

#### 5. An example

In our example the underlying field is  $K = \mathbb{Z}_5$ . We consider the transfer matrix

$$(5.1) \quad W(s) =$$

$$\begin{pmatrix} \frac{2s^6 + 3s^3 + 2s^2 + s + 4}{(s^2 + s + 2)^2(s^3 + 3s^2 + s + 1)} & \frac{s^6 + 4s^3 + s^2 + 2s + 2}{(s^2 + s + 2)^2(s^3 + 3s^2 + s + 1)} \\ \frac{2s^6 + 3s^3 + 2s^2 + s + 1}{(s^2 + s + 2)^2(s^3 + 3s^2 + s + 1)} & \frac{2(3s^6 + 2s^3 + 3s^2 + s + 3)}{(s^2 + s + 2)^2(s^3 + 3s^2 + s + 1)} \end{pmatrix}$$

with entries in  $\mathbb{Z}_5(s)$ . We shall proceed according to the lines of Section 4. The example will allow us to illustrate all aspects of the procedure.

(Step 1.) Partial fraction decomposition of  $W$

Let  $p_1 = s^2 + s + 2$  and  $p_2 = s^3 + 3s^2 + s + 1$ . Then

$$W = W_{p_1} + W_{p_2}$$

and

$$W_{p_1}(s) = \begin{pmatrix} \frac{3s^3 + 4s^2 + s}{(s^2 + s + 2)^2} & \frac{3s^3 + 2s^2 + 3s + 4}{(s^2 + s + 2)^2} \\ \frac{s + 3}{(s^2 + s + 2)^2} & \frac{2s^3 + 4s^2 + 3s}{(s^2 + s + 2)^2} \end{pmatrix}$$

and

$$W_{p_2}(s) = \begin{pmatrix} \frac{4s^2 + 4s + 1}{s^3 + 3s^2 + s + 1} & \frac{3s^2 + 3s + 2}{s^3 + 3s^2 + s + 1} \\ \frac{2s^2 + s + 2}{s^3 + 3s^2 + s + 1} & \frac{4s^2 + 2s + 4}{s^3 + 3s^2 + s + 1} \end{pmatrix}$$

(Step 2.) <sub>$p_1$</sub>  Decomposition of  $W_{p_1}$

The Smith-McMillan form of  $W_{p_1}$  is

$$\Sigma = \text{diag}\left(\frac{a_1}{p^2}, \frac{a_2}{p}\right) = \begin{pmatrix} \frac{1}{(s^2 + s + 2)^2} & 0 \\ 0 & \frac{(s+1)(s^3 + 3s^2 + 4)}{s^2 + s + 2} \end{pmatrix}.$$

We have  $W_{p_1} = U\Sigma V^T$ , and the unimodular matrices  $U$  and  $V$  are given by

$$U = (u_1, u_2) = \begin{pmatrix} s(3s^2 + 4s + 1) & 4s^2 + 3 \\ s + 3 & 3 \end{pmatrix},$$

and

$$V = (v_1, v_2) = \begin{pmatrix} 1 & 0 \\ 2s^5 + 4s^4 + s^3 + 4s^2 + 2 & 1 \end{pmatrix}.$$

Note that

$$\frac{a_2}{p} = \frac{(s+1)(s^3 + 3s^2 + 4)}{s^2 + s + 2}$$

is not strictly proper. We calculate  $\tilde{a}_2$  according to (3.21) and obtain

$$\pi_- \frac{a_2}{p} = \frac{\tilde{a}_2}{p} = \frac{3}{s^2 + s + 2}.$$

Set

$$w_1 = u_1 \frac{a_1}{p^2} v_1^T = \begin{pmatrix} s(3s^2 + 4s + 1) \\ 3 + s \end{pmatrix} \frac{1}{(s^2 + s + 2)^2} \begin{pmatrix} 1 \\ 2s^5 + 4s^4 + s^3 + 4s^2 + 2 \end{pmatrix}^T$$

and

$$w_2 = u_2 \frac{\tilde{a}_2}{p} v_2^T = \begin{pmatrix} 4s^2 + 3 \\ 3 \end{pmatrix} \frac{3}{s^2 + s + 2} \begin{pmatrix} 0 \\ 1 \end{pmatrix}^T.$$

Then  $W_{p_1} = \pi_- w_1 + \pi_- w_2$ .

(Step 2.2.)<sub>w\_1</sub> Realization of  $\pi_- w_1$ .

We set

$$h = u_1 a_1 = \begin{pmatrix} s(3s^2 + 4s + 1) \\ 3 + s \end{pmatrix}$$

and

$$g = v_1 = \begin{pmatrix} 1 \\ 2s^5 + 4s^4 + s^3 + 4s^2 + 2 \end{pmatrix}.$$

Then  $h = h_0 + h_1 p$  with

$$h_0 = \begin{pmatrix} 4s + 3 \\ s + 3 \end{pmatrix}, \quad h_1 = \begin{pmatrix} 3s + 1 \\ 0 \end{pmatrix}.$$

Similarly,  $g = g_0 + g_1 p + g_2 p^2$  with

$$g_0 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \quad g_1 = \begin{pmatrix} 0 \\ s \end{pmatrix}, \quad g_2 = \begin{pmatrix} 0 \\ 2s \end{pmatrix}.$$

This leads to

$$H_0 = \begin{pmatrix} 3 & 4 \\ 3 & 1 \end{pmatrix}, \quad H_1 = \begin{pmatrix} 1 & 3 \\ 0 & 0 \end{pmatrix},$$

and

$$(H_0 \mid H_1) = \left( \begin{array}{cc|cc} 3 & 4 & 1 & 3 \\ 3 & 1 & 0 & 0 \end{array} \right) = H.$$

The matrix  $M$  in (3.1) is given by

$$M = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}.$$

Hence

$$G_0 = \begin{pmatrix} 0 & 1 \\ 0 & 2 \end{pmatrix}, G_1 = \begin{pmatrix} 0 & 0 \\ 1 & -1 \end{pmatrix}$$

such that

$$\left( \frac{G_1^T}{G_0^T} \right) = \begin{pmatrix} 0 & 1 \\ 0 & -1 \\ 0 & 0 \\ 1 & 2 \end{pmatrix} = G.$$

Finally, for  $p = s^2 + s + 2$ , we have

$$J(p^2) = \begin{pmatrix} 0 & 1 & & \\ 3 & 4 & 1 & \\ & 0 & 1 & \\ & & 3 & 4 \end{pmatrix} = F.$$

(Step 2.2)<sub>w2</sub> Realization of  $\pi_- w_2$

Set

$$h = u_2 \tilde{a}_2 = \binom{4s^2 + 3}{3} \times 3 = \binom{2s^2 + 4}{4}$$

and

$$g = v_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Then  $h = h_0 + h_1 p$  with

$$h_0 = \binom{3s}{4}, \quad h_1 = \binom{2}{0}.$$

Hence

$$H_0 = \begin{pmatrix} 0 & 3 \\ 4 & 0 \end{pmatrix} = H.$$

From

$$g_0 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

and (3.8) we obtain

$$G_0 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = G.$$

The corresponding state space matrix is

$$J(p^1) = C(p) = \begin{pmatrix} 0 & 1 \\ 3 & 4 \end{pmatrix} = F.$$

We omit Step 2.3. The pieces of the realization will be together at the end of the procedure.

(Step 2.)<sub>p2</sub> Realization of  $W_{p2}$  The Smith-McMillan form of  $W_{p2}$  is

$$\Sigma = \begin{pmatrix} \frac{1}{s^3 + 3s^2 + s + 1} & 0 \\ 0 & 0 \end{pmatrix}.$$

The unimodular matrices  $U, V$  in the decomposition  $U\Sigma V^T = W_{p_2}$  are

$$U = (u_1, u_2) = \begin{pmatrix} 4s^2 + 4s + 1 & s \\ 2s^2 + s + 2 & 3s + 1 \end{pmatrix}, V = (v_1, v_2) = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}.$$

Set

$$p = s^3 + 3s^2 + s + 1.$$

Then

$$W_{p_2} = u_1 \frac{1}{p} v_1^T = \begin{pmatrix} 4s^2 + 4s + 1 \\ 2s^2 + s + 2 \end{pmatrix} \frac{1}{s^3 + 3s^2 + s + 1} \begin{pmatrix} 1 \\ 0 \end{pmatrix}^T.$$

It is easy to see that one can obtain the minimal realization of  $W_{p_2}$  directly from Lemma 3.1. Note that  $W_{p_2}$  is of the form (3.2) with  $h = u_1$ ,  $g = v_1$ , and  $k = 1$ . Moreover,  $\deg h < \deg p$  and  $\deg g < \deg p$  imply  $H = H_0$  and  $G = G_0$ . Thus  $h = h_0$  yields

$$H_0 = \begin{pmatrix} 1 & 4 & 4 \\ 2 & 1 & 2 \end{pmatrix} = H.$$

From

$$(5.2) \quad M = \begin{pmatrix} 1 & 3 & 1 \\ 3 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

and  $g = g_0$  follows

$$G_0 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 2 \end{pmatrix} = G.$$

Finally, we have

$$J(p^1) = C(p) = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 4 & 4 & 2 \end{pmatrix} = F.$$

### (Step 3.) Realization of $W$

Taking the direct sum of the realizations of  $\pi_- w_1$ ,  $\pi_- w_2$  and  $W_{p_2}$  we obtain

$$F = \left( \begin{array}{ccc|c} 0 & 1 & & \\ 3 & 4 & 1 & \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 3 & 4 \\ \hline & & & \\ & 0 & 1 & \\ & 3 & 4 & \\ \hline & & & \\ & 0 & 1 & 0 \\ & 0 & 0 & 1 \\ & 4 & 4 & 2 \end{array} \right)$$

$$H = \left( \begin{array}{cccc|cc|ccc} 3 & 4 & 1 & 3 & 0 & 3 & 1 & 4 & 4 \\ 3 & 1 & 0 & 0 & 4 & 0 & 2 & 1 & 2 \end{array} \right) \quad \text{and} \quad G = \left( \begin{array}{cc} 0 & 1 \\ 0 & -1 \\ 0 & 0 \\ \hline 1 & 2 \\ 0 & 0 \\ 0 & 1 \\ \hline 0 & 0 \\ 0 & 0 \\ 1 & 2 \end{array} \right).$$

The transfer matrix  $W$  in (5.1) has a minimal realization  $H(sI - F)^{-1}G = W(s)$  where the matrices  $F, G, H$  are the ones displayed above, and  $F$  is in Jacobson normal form.

## References

- [AW] W.A. Adkins and St.W. Weintraub, *Algebra: An Approach via Module Theory*, Springer, New York, 1992.
- [Ay] F. Ayres, Jr., *Schaum's Outline of Theory and Problems of Matrices*, McGraw Hill, New York, 1962.
- [Co] P.M. Cohn, *Algebra*, Vol. 1, Wiley, London, 1974.
- [DJ] J. Della Dora and F. Jung, *Resolvent and rational canonical forms of matrices*. SIGSAM Bull. 30 (117)(1996), 4–10.
- [dS] B. De Schutter, *Minimal state-space realization in linear system theory: An overview*. J. Comput. Appl. Math. 121(2000), 331–354.
- [Fo] G.D. Forney, Jr., *Convolutional codes, I, Algebraic structure*. IEEE Trans. Inform. Theory IT-16(1970), 1970 720–738.
- [Fu] P.A. Fuhrmann, *A Polynomial Approach to Linear Algebra*, Springer, New York, 1996.
- [Gi] A. Gill, *Linear Sequential Circuits*, McGraw-Hill, New York, 1966.
- [IS] N. Ito, W. Schmale, and H.K. Wimmer, *Minimal state space realizations in Jacobson normal form*. Internat. J. Control (to appear).
- [Ja] N. Jacobson, *Lectures in Abstract Algebra, Vol. II - Linear Algebra*, Van Nostrand, Princeton, 1953.
- [Kt] Th. Kailath, *Linear Systems*, Prentice Hall, Englewood Cliffs, 1980.
- [Ka] R.E. Kalman, *Irreducible realizations and the degree of a rational matrix*. J. Soc. Ind. Appl. Math. 13(1965), 520–544.
- [KF] R.E. Kalman, P.L. Falb, and M.A. Arbib, *Topics in mathematical system theory*, McGraw-Hill, New York, 1969.
- [Kr] W. Krull, *Über Begleitmatrizen und Elementarteilertheorie*, Dissertation, Freiburg, 1921. Collected Papers, edited by P. Ribenboim, Bd. 1, pp. 55–95, de Gruyter, Berlin, 1999.
- [KM] K. Kuriyama and Sh. Moritsugu, *A computational method for converting a matrix from rational normal form to Jacobson normal form through computer algebra* (in Japanese). Research Institute for Mathematical Sciences, Kyoto, Kokyuroku 1038(1998), 17–22.
- [MS] J.L. Massey and M.K. Sain, *Codes automata and continuous systems: Explicit interconnections*. IEEE Trans. Automat. Control AC-12(1967), 644–650.
- [Ro] H.H. Rosenbrock, *State-Space and Multivariable Theory*, Wiley, New York, 1970.
- [Rt] J. Rosenthal, *Connections between linear systems and convolutional codes*, in B. Marcus and J. Rosenthal, editors, “Codes, Systems and Graphical Models”, IMA Vol. 123, pp. 39–66, Springer, New York, 2001.
- [RS] J. Rosenthal, J.M. Schuhmacher, and E. Von York, *On behaviors and convolutional codes*. IEEE Trans. Inform. Theory IT-42(1996), 1881–1991.
- [RY] J. Rosenthal and E. Von York, *BCH convolutional codes*. IEEE Trans. Inform. Theory IT-45(1999), 1833–1844.

DEPARTMENT OF MATHEMATICAL EDUCATION, NARA UNIVERSITY OF EDUCATION, TAKABATAKE-CHO, NARA-SHI, NARA 630-8528

*E-mail address:* nucharu@nara-edu.ac.jp

FACHBEREICH 6 MATHEMATIK, CARL-VON-OSSIETZKY-UNIVERSITÄT, D-26111 OLDENBURG, GERMANY

*E-mail address:* wiland.schmale@uni-oldenburg.de

MATHEMATISCHES INSTITUT, UNIVERSITÄT WÜRZBURG, D-97074 WÜRZBURG, GERMANY

*E-mail address:* wimmer@mathematik.uni-wuerzburg.de

# High Order Accurate Particular Solutions of the Biharmonic Equation on General Regions

Anita Mayo

**ABSTRACT.** We present a fast, new method for evaluating particular solutions of the biharmonic equation on general two dimensional regions. The cost of our method is essentially twice the cost of solving Poisson's equation on a regular rectangular region in which the irregular region is embedded. Thus, the cost is  $O(n^2 \log n)$  where  $n$  is the number of mesh points in each direction in the embedding region. Moreover, we can compute derivatives of the particular solution directly, with little loss of accuracy if the boundary and inhomogeneous term are sufficiently accurate. This is especially important in applications, since it is generally derivatives that are needed. Computational results are provided.

## 1. Introduction

In this paper we present a rapid new fourth order accurate method for evaluating particular solutions of the biharmonic equation on general two dimensional regions. That is, given a function  $f$  defined on a region  $D$  we show how to find a function  $V$  such that  $\Delta\Delta V = f$  on  $D$ .

Our method uses integral representations of the particular solution and its derivatives. Specifically, we use the fact that any particular solution can be represented as the volume integral of the product of the inhomogeneous term  $f$  and a fundamental solution of the biharmonic equation. In order to evaluate such an integral we use a rapid fourth order accurate difference method we have developed. It is an extension of a method we developed previously [M2], [G2], [Mc], for evaluating fundamental solutions of Poisson's equation on general regions. (We note that other fast methods have been developed for solving Poisson's equation on general regions. See, for example, [A2].)

It is particularly convenient to have an integral formulation of the particular solution when using integral equation methods to solve an inhomogeneous differential equation, especially when the problem domain is an exterior region. Once a particular solution has been evaluated, the computational problem is reduced to the solution of a homogeneous differential equation, which only requires the solution of an integral equation on the surface of the region.

---

1991 *Mathematics Subject Classification.* Primary 65E05, 65R20.

*Key words and phrases.* biharmonic equations, integral equations, finite difference method.

Our method for evaluating particular solutions uses rapid methods of solving Laplace's equation, for example [A], [Bu], on a regular rectangular region with a rectangular mesh in which the general region of integration is embedded. In fact, the number of operations needed to evaluate the integral is basically equal to twice the number of operations needed to solve Poisson's equation on a regular rectangular grid using a fast Poisson solver. This is despite the fact that the region  $D$  over which the integration is performed is irregular. Also, one need not extend the inhomogeneous term  $f$  as a smooth function.

A standard way to evaluate a particular solution of a differential equation is by using a quadrature formula, but this can be very expensive. Specifically, a total of  $O(n^4)$  operations are needed to evaluate the integral at every point of an  $n$  by  $n$  grid, since evaluating each integral requires  $O(n^2)$  operations. In contrast, our method only requires  $O(n^2 \log n)$  operations.

Another problem with using quadrature formulas in a straightforward way is that fundamental solutions of the biharmonic equation have discontinuities in their derivatives when their arguments approach 0, that is as the point at which one is evaluating the integral nears a point of the region of integration. This is due to the fact that the functions  $\log r$  and  $r$  have unbounded derivatives at the origin. So, it is difficult to compute these integrals very accurately at points in, or near, the region of integration.

The method we use does not have these problems.

We note that it is also possible to evaluate a particular solution of the biharmonic equation using finite element methods.

There are, however, several difficulties associated with these methods as well. First, they are not well suited to exterior regions. Second, since the matrix equation that needs to be solved has condition number proportional to  $n^4$ , the convergence of iterative solution methods tends to be very slow. Direct methods of solving the matrix equation, such as nested dissection require  $O(n^3)$  operations with a large constant. See [S]. Also, although very rapid finite difference methods are available for solving the biharmonic equation on a region, see [B], [G], they require the regions to be rectangular.

The essential idea of our method is the following. We first evaluate the Laplacian  $W$  of the particular solution  $V$ . Such a function satisfies the inhomogeneous Poisson equation  $\Delta W = f$  on  $D$ , and can therefore be expressed as (volume) integral of the product of  $f$  and a fundamental solution of Poisson's equation, such as  $\log r$ . We evaluate an approximation to  $W$  by first embedding  $D$  in a larger rectangular region  $R$  with a uniform mesh, and computing a fourth order accurate approximation to the discrete Laplacian of  $W$  at all the mesh points of  $R$ . Once we have done this, we apply a fast Poisson solver on  $R$  to obtain an approximation to  $W$ . It is easy to compute an approximation to the discrete Laplacian of  $W$  at most mesh points of  $R$ . At mesh points inside  $D$  that have all their neighboring mesh points inside  $D$  we approximate the discrete Laplacian by  $f$  since  $\Delta W = f$  on  $D$ . Similarly, at mesh points outside  $D$  whose neighboring mesh points are also outside  $D$ , we set the discrete Laplacian equal to zero since  $\Delta W = 0$  outside  $D$ . The difficulty arises at the other, "irregular" mesh points. Because of the discontinuities in the second derivatives of  $W$  across the boundary of  $D$  the discrete Laplacian is not well approximated by the continuous Laplacian. It turns out, however, that one can compute an approximation to the discrete Laplacian using the discontinuities

in  $W$ , which depend on  $f$  and its derivatives, and information about the boundary of  $D$ .

Once we have an approximation to  $W$  we compute an approximation to  $V$ , again by computing an approximation to its discrete Laplacian, and applying a fast Poisson solver. Since  $\Delta V = W$ , and an approximation to  $W$  is known, the problem of computing an approximation to the discrete Laplacian of  $W$  is only difficult at the irregular mesh points. Again, we can compute an approximation at such points in terms of certain derivatives of  $f$  and information about the boundary of  $D$ .

The organization of this paper is as follows. In Section 2 we show how to compute a particular solution of the biharmonic equation, and in Section 3 we show how to compute derivatives of the particular solution directly and describe an application. In Section 4 we provide results of numerical experiments.

## 2. Evaluation of Particular Solutions of the Biharmonic Equation

In this section we describe our method for evaluating an integral  $V(x, y)$  whose kernel is a fundamental solution of the biharmonic equation, for example, an integral of the form

$$(2.1) \quad V = \int_D \int (r^2(\log r - 1) + g) f dx dy$$

where  $g$  is any function such that  $\Delta\Delta g = 0$ .

We later describe how to evaluate the derivatives of such an integral.

Initially we ignore boundary conditions since we only seek a particular solution of the biharmonic equation.

We first note that the derivatives of order less than or equal to three of an integral like (2.1) are continuous across the boundary of the region  $D$ . Its fourth derivatives are, of course, discontinuous since

$$(2.2) \quad \Delta\Delta V = f \text{ in } D \text{ and } \Delta\Delta V = 0 \text{ outside } D.$$

The Laplacian of the integral,

$$(2.3) \quad \Delta V = W = \int_D \int f \log r dx dy,$$

known as a Newton potential, satisfies Poisson's equation.

$$(2.4) \quad \Delta W = f \text{ in } D \text{ and } \Delta W = 0 \text{ outside } D.$$

We first show how to compute an approximation,  $W_h$ , to such an integral. For simplicity we begin by showing how to compute a second order accurate approximation.

We start by embedding the irregular region  $D$  over which we are evaluating the integral in a larger rectangular region  $R$  with a uniform grid which ignores the boundary of  $D$ .

We evaluate this integral by computing an approximation to its 5 point discrete Laplacian

$$\Delta_h W = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} W$$

at all the mesh points of the grid. Once we have done that, we apply an operator, known as a Poisson solver, which inverts the 5 point discrete Laplacian on  $R$ . A Poisson solver is merely an algorithm for inverting the discrete Laplacian. Therefore, if we prescribe the discrete Laplacian of a function at every mesh point of  $R$ , and apply a fast Poisson solver, then we will have an approximation to the original function, whether or not it is smooth. We note that applying a Poisson on a grid with  $n$  points only requires  $O(n^2 \log n)$  operations [Bu].

Using (2.4), at mesh points inside  $D$ , which have all their neighboring mesh points inside  $D$  we set  $\Delta_h^5 W_h = f$ , and at points outside  $D$ , with all their neighbors outside we set  $\Delta_h^5 W_h = 0$ . The accuracy of these equations is the same as the accuracy of the discrete Laplacian, that is, they are second order accurate.

The problem then reduces to computing an approximation to the discrete Laplacian at the other mesh points, the points which are in one region, but have neighboring mesh points in the other region. Since, (by 2.4), the integral  $W$  isn't smooth across the boundary of  $D$ , we cannot use either formula at these points.

It turns out that in order to be able to compute an approximation to  $\Delta_h^5 W$  at these points it is sufficient to know what the discontinuities in the derivatives of  $W$  in the coordinate directions are at the boundary of the region  $D$ . We now show how to find these discontinuities.

Suppose the boundary of  $D$  is given by  $(x(t), y(t))$ . For a given function  $g$  defined on  $R$  which is discontinuous across  $\partial D$  let  $[g(p)]$  denote the discontinuity in  $g$  at a point  $p$  on  $\partial D$ . That is, it is the difference between the limiting values as  $p$  is approached from inside and outside  $D$ . An integral of the form (2.3) and its normal derivative are continuous across  $\partial D$ . Therefore, for  $p$  in  $\partial D$

$$(2.5) \quad [W(p)] = 0,$$

and

$$(2.6) \quad [W_n(p)] = \dot{y}(t)[W_x(p)] - \dot{x}(t)[W_y(p)] = 0.$$

By differentiating (2.5) in the tangential direction  $t$ , we see that

$$(2.7) \quad [W_t(p)] = \dot{x}(t)[W_x(p)] + \dot{y}(t)[W_y(p)] = 0.$$

Equations (2.6) and (2.7) imply that at all points  $p \in \partial D$

$$(2.8) \quad [W_x(p)] = 0, \text{ and } [W_y(p)] = 0.$$

To find the discontinuities in higher order derivatives of  $W$  we use (2.4) and the above four equations.

For example, to find discontinuities in the second derivatives of  $W$  we differentiate (2.6) and (2.7) in the tangential direction and use (2.8) to obtain

$$(2.9) \quad \dot{x}(t)^2[W_{xx}] + \dot{y}(t)^2[W_{yy}] + 2\dot{x}(t)\dot{y}(t)[W_{xy}] = 0.$$

$$(2.10) \quad \dot{x}(t)\dot{y}(t)[W_{xx}] - \dot{x}(t)\dot{y}(t)[W_{yy}] + (\dot{y}^2(t) - \dot{x}^2(t))[W_{xy}] = 0.$$

We also note that

$$(2.11) \quad [W_{xx}] + [W_{yy}] = f.$$

Equations (2.9), (2.10) and (2.11) form a three by three linear system of equations that can be used to solve for  $[W_{xx}]$ ,  $[W_{yy}]$  and  $[W_{xy}]$  at any point on the boundary of  $D$ . The determinant of the system,  $(\dot{x}(t)^2 + \dot{y}(t)^2)^2$ , is nonzero at all points, and, therefore, the equations have a unique solution.

We use similar methods to evaluate discontinuities in the third and higher order derivatives of  $W$ . To determine the discontinuities in these derivatives we differentiate equations (2.9) and (2.10) in the tangential  $t$  direction, and we differentiate (2.11) in both the normal and tangential directions. This gives us four equations to solve for the discontinuities in the four third derivatives of  $W$ . We can continue using this method to compute discontinuities in higher order derivatives.

Using the above equations we can compute the discontinuity in any mixed normal and tangential derivative of  $W$ . These discontinuities can, in turn, be used to compute the discontinuities in any derivative of  $W$  in the  $x$  and  $y$  coordinate directions.

We now show how to use this information to obtain an approximation to the discrete Laplacian of  $W$  at points near the boundary of  $D$ .

We let  $\tilde{w}(p)$  denote the values of  $W(p)$  at points  $p$  outside  $D$ , we let  $w(p)$  denote the values of  $W(p)$  at points  $p$  inside  $D$ , and we let  $B$  be the set of irregular mesh points, that is, the set of points which have one of their neighboring mesh points on the opposite side of  $\partial D$ .

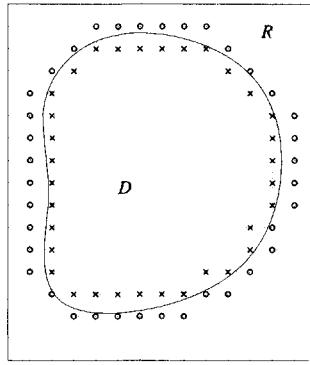


FIGURE 1

Suppose, for example,  $p$  is in  $D$ , but its neighbor to the right  $p_E$ , is not. Let  $p^*$  be the point on the line between  $p$  and  $p_E$  which intersects  $\partial D$ , let  $h_1$  be the distance between  $p$  and  $p^*$ , and let  $h_2 = h - h_1$ .

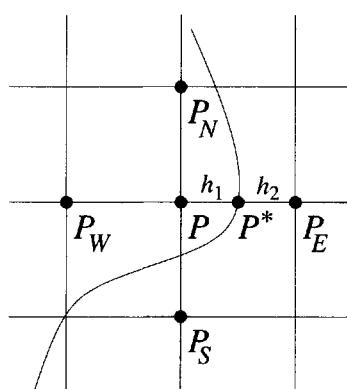


FIGURE 2

By manipulating the Taylor series at  $p$  and  $p_E$  both evaluated at  $p^*$ , we can derive the following expression for  $W(p_E) - W(p)$  (For details see [M]).

$$(2.12) \quad \begin{aligned} \tilde{w}(p_E) - w(p) &= [\tilde{w}(p^*) - w(p^*)] + h_2[\tilde{w}_x(p^*) - w_x(p^*)] + \frac{1}{2}h_2^2[\tilde{w}_{xx}(p^*) - w_{xx}(p^*)] \\ &\quad + \frac{1}{6}h_2^3[\tilde{w}_{xxx}(p^*) - w_{xxx}(p^*)] + hw_x(p) + \frac{1}{2}h^2w_{xx}(p) + \frac{1}{6}h^3w_{xxx}(p) + O(h^4) \end{aligned}$$

Note that the first four terms on the right hand side depend on the discontinuities in  $W$  and in its  $x$  derivatives at the boundary of  $D$ . The other terms are the usual Taylor series terms. Therefore, the right hand side of (2.12) is a sum of terms we can evaluate in terms of the discontinuities in  $W$  and its  $x$  derivatives, and terms we would have if the boundary of  $D$  did not pass between  $p$  and  $p_E$ .

Now let  $p_W, p_N, p_S$  be the mesh points to the left of, above, and below  $p$ . We get the same type of expressions for the differences between the value of  $W$  at  $p$  and at its other neighbors, that is  $W(p_W) - W(p)$ ,  $W(p_N) - W(p)$ ,  $W(p_S) - W(p)$ , except that there will not be any boundary terms unless  $\partial D$  passes between  $p$  and that neighbor. Therefore, we can compute an approximation to the 5 point discrete Laplacian of  $W$ , which is just the sum of the above four differences divided by  $h^2$

$$\Delta_h^5 W(p) = \frac{W(p_E) + W(p_W) + W(p_N) + W(p_S) - 4W(p)}{h^2}.$$

at all the irregular points.

More precisely, for mesh points  $B$  we define the mesh function  $m$  to be the value of the extra terms in the discrete Laplacian due to the discontinuities in  $W$  and its derivatives.

We define  $W_h$  to be the solution of the following equations:

$$\Delta_h^5 W_h(p) = \begin{cases} f(p) & p \in D - B \\ f(p) + m(p) & p \in B \cap D \\ m(p) & p \in B \cap D^c \\ 0 & p \in R - D \cap B \end{cases}$$

If the values of  $m(p)$  are third order accurate, then by applying a second order accurate Poisson solver we obtain a second order accurate approximation  $W_h$  to  $W$ . See [M], [M2].

We can clearly also compute higher order accurate approximations to  $W$  by using higher order accurate approximations to the Laplacian.

Specifically, if  $\Delta f$  is known, we can replace  $\Delta_h^5 W$  by

$$\Delta_h^9 W = \frac{1}{6h^2} \begin{bmatrix} 1 & 4 & 1 \\ 4 & -20 & 4 \\ 1 & 4 & 1 \end{bmatrix} W$$

The equation  $\Delta W = f$  is approximated by

$$\Delta_h^9 W = f + h^2 \Delta f + O(h^4)$$

We can write

$$\Delta_h^9 = \frac{2}{3h^2} \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} + \frac{1}{3h^2} \begin{bmatrix} 1 & 0 & 1 \\ 0 & -4 & 0 \\ 1 & 0 & 1 \end{bmatrix} = \frac{2}{3} \Delta_h^5 + \frac{1}{3} \Delta_h^x$$

where  $\Delta_h^x$  is a rotation of  $\Delta_h^5$  by 45 degrees. This means that the jump relations which need to be used for evaluating  $\Delta_h^x W$  are the same as for  $\Delta_h^5$  after rotating the axes by 45 degrees, but keeping  $D$  fixed. Thus, this higher accuracy does not require any additional effort. Once we have an approximation to the 9 point discrete Laplacian of  $W$ , by applying a fourth order accurate Poisson solver which uses this discrete Laplacian we obtain a fourth order accurate solution.

Once we have an approximation to  $W$  we can compute an approximation to  $V$ . We compute the approximation by first computing an approximation to  $\Delta V$  at all mesh points of  $R$ , and then applying a Poisson solver. To approximate  $\Delta_h W$  at the regular points we use the fact that  $\Delta V = W$ , and so we set  $\Delta_h V_h = W_h$ . In order to approximate  $\Delta_h V$  at the irregular mesh points we need to find the discontinuities in the fourth derivatives of  $V$ .

This is easy to do. To compute the discontinuities in the five fourth derivatives we use the five equations

$$(2.13) \quad [V_{tttt}] = 0, \quad [V_{tttn}] = 0, \quad [V_{ttnn}] = 0,$$

$$[V_{tnnn}] = 0, \quad [\Delta\Delta V] = f.$$

Before applying a Poisson solvers we, of course, need to prescribe boundary conditions. If we only want a particular solution of Poisson's or the biharmonic equations, then there is no difficulty, since the discontinuities in the derivatives of the integrals are the same, and therefore the discrete Laplacians, will be the same, independent of which fundamental solution of the biharmonic is used as the kernel. The integral we obtain an approximation to with this method is the one associated with the same boundary conditions as the fast Poisson solver we use. For example, if we use a doubly periodic Poisson solver, then we obtain an approximation to the integral whose kernel is the doubly periodic Greens function for the biharmonic equation. If we need an integral with a specific kernel, then we use the corresponding Poisson solver.

### 3. Computation of Derivatives and Applications

We can essentially use the method we have described to also compute components of the derivatives of  $W$  and  $V$  directly. For example, we can approximate the integral

$$W_x(x, y) = \int_D \int f \frac{\partial \log r}{\partial x} dx dy$$

This follows because we know  $\Delta W_x$ ; it is  $\frac{\partial f}{\partial x}$  inside  $D$ , and zero outside. Thus, we can approximate the discrete Laplacain of  $W_x$  at the regular mesh points. Furthermore, we can compute the discontinuities in all derivatives of  $W_x$  at the boundary of  $D$ , since we know the discontinuities in the derivatives of  $W$ . As before, once we know these discontinuities we can use them to compute an approximation to the discrete Laplacian of  $W_x$ . We then apply a Poisson solver to obtain an approximation to  $W_x = (\Delta V)_x$ . Similarly, we can approximate  $W_y = (\Delta V)_y$ . Again, the accuracy of the method is same as the accuracy of the discrete Laplacian and the accuracy to which the discrete Laplacian is computed.

Once we have a fourth order accurate approximations to  $W_x$  and  $W_y$  we can find approximations to  $V_x$  and  $V_y$ . We also do this by computing fourth order accurate approximations to  $\Delta_h^9 V_x$  and  $\Delta_h V_y$  and applying a Poisson solvers. Since we know

$\Delta V_x = W_x$  and  $\Delta V_y = W_y$ , the problem reduces to evaluating the discontinuities in the derivatives of  $V_x$  and  $V_y$  in the coordinate directions.

This can be done using (2.13). Again, once we know the discrete Laplacians of the functions we need only apply Poisson solvers to obtain approximations to the functions.

The methods we have described for computing particular solutions of the biharmonic equation and their derivatives can be used for solving more difficult problems.

For example, suppose we seek a solution of the two dimensional Navier Stokes equations inside or outside a bounded region  $D$ :

$$(3.1) \quad uu_x + vu_y = \nu \Delta u - \frac{1}{\rho} p_x$$

$$(3.2) \quad uv_x + vv_y = \nu \Delta v - \frac{1}{\rho} p_y$$

$$(3.3) \quad u_x + v_y = 0$$

Here  $u$  and  $v$  are the velocity components,  $p$  is the pressure,  $\rho$  is the fluid density and  $\nu$  is the coefficient of viscosity. See [L], [Mi] [Mil]. Since the fluid is incompressible, one can obtain the fluid velocity from a stream function. That is, by (3.3) there is a function  $\Phi(x, y)$  such that

$$(3.4) \quad \Phi_y = u \text{ and } \Phi_x = -v.$$

Substituting these equations in (3.1) and (3.2) and eliminating the pressure  $p$ , we see

$$(3.5) \quad \Delta^2 \Phi = \frac{1}{\nu} (\Phi_y(\Delta \Phi)_x - \Phi_x(\Delta \Phi)_y)$$

So,  $\Phi(x, y)$  satisfies an inhomogeneous biharmonic equation whose right hand side is nonlinear and depends on the components of the gradient of the stream function and on the components of the gradient of the Laplacian of the stream function.

We can solve this equation when the Reynolds number is sufficiently small by Picard iteration. See [Mil], [P]. Specifically, at each iteration we solve the inhomogeneous biharmonic equation with gradient specified on the boundary of the region.

$$(3.6) \quad \Delta^2 \Phi^{n+1} = \frac{1}{\nu} (\Phi_y^n(\Delta \Phi^n)_x - \Phi_x^n(\Delta \Phi^n)_y) \\ \nabla \Phi^n = (g_1, g_2) \text{ on } \partial D$$

When using an integral equation formulation to solve such an inhomogeneous equation one first computes a particular solution

$$V^n = \iint_D F^n G dx dy,$$

where  $G$  is a fundamental solution of the biharmonic equation and

$$F^n = \frac{1}{\nu} (\Phi_y^n(\Delta \Phi^n)_x - \Phi_x^n(\Delta \Phi^n)_y)$$

is the right hand side of (3.6).

Once we have evaluated the function  $V^n$  and its gradient

$$\nabla V^n = \iint_D F^n \nabla G dx dy,$$

the problem is reduced to solving a homogeneous biharmonic equation with prescribed boundary conditions:

$$\Delta\Delta U^n = 0$$

$$(U_x^n, U_y^n) = (g_1 - V_x^n, g_2 - V_y^n) \text{ on } \partial D$$

Very rapid methods are available for solving the homogeneous biharmonic equation. See [G] [Gr] [M3].

After computing the function  $U^n$  we set

$$\Phi^{n+1} = V^n + U^n.$$

#### 4. Results of numerical experiments.

In this section we present results of numerical experiments we performed on an IBM 3090 computer using double precision arithmetic.

We chose two problems for which one can evaluate the integrals (2.1) and their derivatives analytically, both inside and outside the region  $D$ .

Specifically, when  $D$  is the disc of radius  $r_0$  and  $f = \frac{2(2-r_0^2)}{3r_0^2}$  one can evaluate the integral (2.1) when  $g = r_0^2 \log r$ . That is, one can show

$$V(r) = ar^4 + br^2 + c \text{ inside } D$$

and

$$V(r) = r^2(\log r - 1) + \frac{r_0^2}{2} \log r \text{ outside } D$$

where  $a = \frac{2-r_0^2}{24r_0^2}$ ,  $b = \frac{r_0-1+2r_0 \log r_0}{r_0}$ , and  $c = \frac{r_0^2}{2} \log r_0 - ar_0^4 - (b-1)r_0^2$ .

For our calculations we embedded the region  $D(r) = |r| \leq .2$  in a square of sides 1.6 centered at the origin, and we evaluated  $V$ ,  $V_x$  and  $V_y$ . We note that we used a Buneman solver as the Poisson solver. Results are given in Table 1.

In the table  $n$  is the number of grid points in each direction on the square, and the numbers in columns 2, 3 and 4 are the maximum errors we obtained in computing the functions.

TABLE 1

$n$	err. in $V$	err. in $V_x$	err. in $V_y$
16	.61E - 02	.23E - 01	.28E - 01
32	.16E - 03	.96E - 03	.76E - 03
64	.12E - 05	.74E - 04	.78E - 04
128	.22E - 07	.31E - 06	.49E - 06

These numbers show that our method is indeed fourth order accurate. We also note that we experimented with different embedding regions and regions  $D$ , and obtained essentially the same degree of accuracy. (The errors depend on the discretization errors.)

Next we choose the same irregular region  $D$  and embedding region  $R$ , but choose the inhomogeneous term  $f$  so that

$$V(r) = ar^6 + br^4 + cr^2 + d \text{ inside } D$$

and

$$V(r) = r^2 (\log r - 1) \text{ outside } D$$

where  $a = -\frac{1}{12r_0^4}$ ,  $b = \frac{1}{r_0^2}$ ,  $c = -\frac{1}{4} + \frac{\log(r_0)}{2}$  and  $d = r_0^2 \log r_0 - ar_0^6 - br_0^4 - cr_0^2$

For this example we computed  $\Delta V = V_{xx} + V_{yy}$ ,  $\Delta V_x$ , and  $\Delta V_y$ , and we report the maximum relative error. Results are shown in the next table.

TABLE 2

$n$	err. in $\Delta V$	err. in $\Delta V_x$	err. in $\Delta V_y$
32	.83E - 01	.95E - 00	.86E - 00
64	.12E - 02	.71E - 01	.80E - 01
128	.29E - 03	.44E - 02	.65E - 02
256	.20E - 04	.11E - 02	.99E - 03

## References

- [A] A. Averbuch, M. Israeli and L. Vozovoi, *A fast Poisson solver of arbitrary order accuracy in rectangular regions*, SIAM J. Sci. Comput., 19, 1998, pp. 933-952.
- [A2] A. Averbuch, E. Braverman and M. Israeli, *Parallel adaptive solution of a Poisson equation with multiwavelets*, SIAM J. Sci. Comput., 22, 2000, pp. 1053-1086.
- [B] P. Bjorstad, *Fast numerical solution of the biharmonic Dirichlet problem on rectangles*, SIAM J. on Numer. Anal., 20, 1983, pp. 59-81.
- [Bu] O. Buneman, *A compact noniterative Poisson solver*, Rep. SUIPR-294, Inst. Plasma Research, Stanford University, Stanford CA, 1969.
- [Bu2] B. Buzbee and F. Dorr, *The discrete solution of the biharmonic equation on rectangular regions and the Poisson equation on irregular regions*, SIAM J. Numer. Anal., 11, pp. 753-763.
- [G] A. Greenbaum, L. Greengard, and A. Mayo, *On the numerical solution of the biharmonic equation in the plane*, Physica D, 60, 1992, pp. 216-225.
- [G2] A. Greenbaum and A. Mayo, *Rapid parallel evaluation of three dimensional integrals in potential theory*, J. Comput. Phys., 145, 1998, pp. 145-164.
- [Gr] L. Greengard, M Kropinski, A. Mayo, *Integral equation methods for Stokes flow and isotropic elasticity in the plane*, J. of Comput. Phys., 125, (1996), pp. 403-426.
- [J] M. Jaswon and G. Symm, *Integral Equation Methods in Potential Theory and Elastostatics*, Academic Press, New York, 1977.
- [L] O. Ladyzhenskaya, *The Mathematical Theory of Viscous Incompressible Flow*, Gordon and Breach, New York, 1969.
- [La] W. Langlois, *Slow Viscous Flow*, Macmillan, New York, 1964.
- [M] A. Mayo, *The fast solution of Poisson's and the biharmonic equations on general regions*, SIAM J. Numer. Anal., 21, 1984, pp. 285-299.
- [M2] A. Mayo, *The rapid evaluation of volume integrals of potential theory on general regions*, J. Comput. Phys., 100, 1992, pp. 236-245.
- [M3] A. Mayo and A. Greenbaum, *Fast parallel solution of Poisson's and the Biharmonic equations on irregular regions*, SIAM J. on Sci. and Stat. Comput., 13, 1992, pp. 1-17.
- [Mc] A. McKenney, L. Greengard and A. Mayo, *A fast Poisson solver for complex geometries*, J. Comput. Phys., 118 1995, pp. 348-355.
- [Mi] S. G. Mikhlin, *Integral Equations and their Applications*, Pergamon, London, 1957.
- [Mil] R.D. Mills, *Computing internal viscous flow problems for the circle by integral methods*, J. Fluid Mech., 79, (1977) 609-624.
- [P] C. Pozrikidis, *Boundary Integral and Singularity Methods for Linearized Viscous Flow*, Cambridge University Press, Cambridge, 1992.

- [S] A. Sherman, *On the efficient solution of sparse systems of linear and nonlinear equations*, Ph.D. Thesis, Dept. of Comp. Sci., Yale University, 1975.

WATSON RESEARCH CENTER, YORKTOWN HTS., NY 10598  
E-mail address: [amayo@cs.nyu.edu](mailto:amayo@cs.nyu.edu)

*This page intentionally left blank*

## A Fast Projected Conjugate Gradient Algorithm for Training Support Vector Machines

Tong Wen, Alan Edelman, and David Gorsich

**ABSTRACT.** Support Vector Machines (SVMs) are of current interest in the solution of classification problems. However, serious challenges appear in the training problem when the training set is large. Training SVMs involves solving a linearly constrained quadratic programming problem. In this paper, we present a fast and easy-to-implement projected Conjugate Gradient algorithm for solving this quadratic programming problem.

Compared with the existing ones, this algorithm tries to be adaptive to each training problem and each computer's memory hierarchy. Although written in a high-level programming language, numerical experiments show that the performance of its MATLAB implementation is competitive with that of benchmark C/C++ codes such as SVM<sup>light</sup> and SvmFu. The parallelism of this algorithm is also discussed in this paper.

### Introduction

Support Vector Machines (SVMs) have attracted recent attention as a technique to attack classification problems, where prior statistical knowledge about the underlying classes is not available. Two examples are the face detection problem [13] [18] and the handwritten-digit recognition problem [4] [14] [15] [16]. In summary, SVMs tell the difference between two classes by learning from examples of these two classes. The learning procedure involves solving a linearly constrained quadratic programming problem, which is challenging because its size can be large. In this paper, we present a fast and easy-to-implement projected Conjugate Gradient algorithm for solving this quadratic programming problem.

To establish our notation, we first give a brief introduction to SVMs in Section 1. We then present this algorithm in the language of linear algebra in Section 2. Also in this section, addressed are the issues of how to be adaptive to each training problem and each computer's memory architecture. The MATLAB implementation of this algorithm is compared numerically with two benchmark C/C++ codes SVM<sup>light</sup> and SvmFu in Section 3. The timing results based on two representative

---

2000 *Mathematics Subject Classification.* Primary 65F10 90C20; Secondary 65Y20 68T10.

This paper is supported by U.S. Army TACOM under the contract TCN 00-130 awarded by Battelle-Research Triangle Park and NSF under the grant DMS-9971591.

We would like to thank Ryan Rifkin at M.I.T. for many valuable discussions.

training sets of size  $O(10,000)$  show that the performance of this MATLAB implementation is very competitive. Parallelism is discussed in Section 4. Finally, we conclude this paper with Section 5 exploring future work.

## 1. Support Vector Machines

The goal of this introduction is to set the ground for our later discussion. For more exhaustive treatments of SVMs, we refer readers to [5] [6] [21] [24]. As a matter of notation, we use bold typeface for vectors, and normal typeface for scalars such as vector and matrix components. Matrices are indicated by capital letters.

SVMs tell the difference between two classes by learning from instances or examples of these two classes. Geometrically, SVMs estimate the optimal separating boundary between these two classes by a pair of parallel hyperplanes. If the training examples of these two classes are linearly separable as indicated in Figure 1, the maximal-margin SVM hyperplanes are the pair that separates these training examples with the maximum gap. Each training example is represented by a pair  $(\mathbf{x}_i, y_i)$ , where  $\mathbf{x}_i \in \mathbb{R}^n$  and  $y_i \in \{-1, 1\}$ , for  $i = 1, \dots, m$ . The vector  $\mathbf{x}_i$  is the description of the  $i$ th example, while the scalar  $y_i$  is the label indicating which class this example belongs to. Let  $\mathbf{w}$  be the vector defining the normal direction of an oriented hyperplane. We can represent any two parallel (oriented) hyperplanes by the following equation:

$$(1.1) \quad \mathbf{w}^T \mathbf{x} + b = \pm 1, \text{ where } b \in \mathbb{R}.$$

It is easy to see that the distance between these two parallel hyperplanes is

$$(1.2) \quad d = \frac{2}{\|\mathbf{w}\|_2}.$$

For  $i = 1, \dots, m$ , the following inequality enforces that the hyperplanes must separate all the training examples:

$$(1.3) \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1.$$

Therefore, if the training examples are linearly separable, the pair of maximal-margin hyperplanes can be computed by solving the following linearly constrained quadratic programming (QP) problem:

$$(1.4) \quad \underset{\mathbf{w}, b}{\text{minimize}} \quad f(\mathbf{w}, b) \equiv \frac{1}{2} \|\mathbf{w}\|_2^2$$

subject to

$$(1.5) \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, \quad \text{for } i = 1, \dots, m.$$

The assumption of linear separability in Figure 1 is not general, because given  $m$  points in  $\mathbb{R}^n$ , they may not be separable by hyperplanes. To separate linearly non-separable examples, a map  $\phi(\cdot)$  is used to map them to a higher dimensional space  $\mathbb{R}^{n'}$  ( $n' > n$ ), so that in  $\mathbb{R}^{n'}$ , the mapped examples can be separated by a pair of parallel hyperplanes. Note that our hyperplane model still holds here. The only difference is that  $\mathbf{x}_i$  is replaced by  $\phi(\mathbf{x}_i)$ . The underlying idea is simple: if a linear function is not enough to separate the examples, then a nonlinear one is used. Since only inner products are involved in computing the SVM separating hyperplanes as we will see later, a positive definite kernel function  $k(\cdot)$  is used instead of  $\phi(\cdot)$ , where

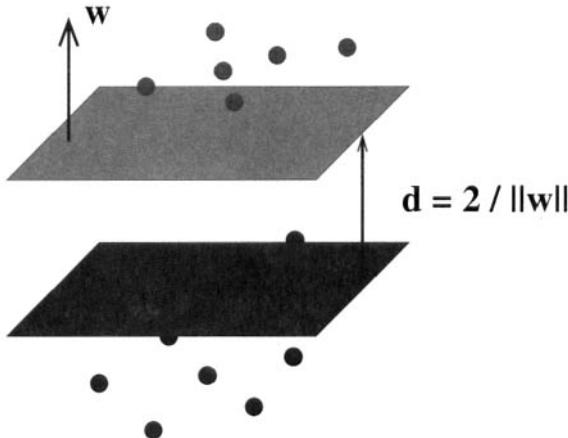


FIGURE 1. A pair of parallel (oriented) hyperplanes separating examples of two classes labeled by  $\pm 1$ , where point  $\mathbf{x}_i$  along with its label  $y_i$  represents one training example  $(\mathbf{x}_i, y_i)$ .  $\mathbf{w}$  gives the normal direction of these two hyperplanes, which points to the positive (gray) points. The positive and the negative (dark) points are linearly separable.

$k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ . The kernel function  $k(\cdot)$  is preferred due to the curse of dimensionality.

Given  $m$  training examples, we can always separate them by choosing a certain kernel function  $k(\cdot)$ . However, to avoid overfitting we may not separate the training examples exactly. Remember that the goal here is to distinguish the two classes as accurately as possible rather than to separate a particular set of training examples correctly. To incorporate the case where examples are not exactly separable, error terms (nonnegative slack variables)  $\varepsilon_i$  are introduced. The generalized QP problem is:

$$(1.6) \quad \underset{\mathbf{w}, b, \varepsilon}{\text{minimize}} \quad f(\mathbf{w}, b, \varepsilon) \equiv \frac{1}{2} \|\mathbf{w}\|_2^2 + c \sum_i \varepsilon_i$$

subject to

$$(1.7) \quad y_i (\mathbf{w}^T \phi(\mathbf{x}_i) + b) \geq 1 - \varepsilon_i \text{ and } \varepsilon_i \geq 0, \text{ for } i = 1, \dots, m.$$

The pair of hyperplanes determined by the above problem is referred as the soft-margin SVM hyperplanes.

If the training point  $\phi(\mathbf{x}_i)$  is correctly separated by the soft-margin hyperplanes, then  $\varepsilon_i = 0$ . To make the SVM hyperplanes represent the general difference between two classes, a large gap (a small  $\|\mathbf{w}\|_2$ ) is preferred, whereas the goal to separate the training examples correctly (the goal to minimize the error term  $\sum_i \varepsilon_i$ ) tends to reduce the gap. The constant  $c$  (specified beforehand) determines the trade-off between the size of the gap and how well the examples are separated.

Problem (1.6) subject to (1.7) is referred as the primal SVM QP problem, which as we have seen has a clear geometric meaning. However, in practice its dual

problem is solved because of its simplicity:

$$(1.8) \quad \underset{\alpha}{\text{maximize}} \quad F(\alpha) \equiv \alpha^T \mathbf{1} - \frac{1}{2} \alpha^T H \alpha$$

subject to

$$(1.9) \quad \mathbf{y}^T \alpha = 0,$$

$$(1.10) \quad \mathbf{0} \leq \alpha \leq \mathbf{c},$$

where  $\alpha = [\alpha_1, \dots, \alpha_m]^T$ ,  $\mathbf{y} = [y_1, \dots, y_m]^T$ ,  $\mathbf{1} = [1, \dots, 1]^T$  and  $\mathbf{c} = c\mathbf{1}$ .  $H$  is a  $m \times m$  symmetric positive semi-definite matrix with

$$\begin{aligned} H_{ij} &= y_i (\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)) y_j \\ &= y_i k(\mathbf{x}_i, \mathbf{x}_j) y_j. \end{aligned}$$

Each dual variable (Lagrange multiplier)  $\alpha_i$  corresponds to one example  $(\mathbf{x}_i, y_i)$ . To simplify our notation, we use  $\mathbf{x}_i$  in place of  $\phi(\mathbf{x}_i)$  in our remaining discussion, considering that the hyperplane model always holds. At optimality, the primal variables  $\mathbf{w}$  and  $b$  are determined by  $\alpha$  as the following:

$$(1.11) \quad \mathbf{w} = \sum_{i=1}^m y_i \alpha_i \mathbf{x}_i,$$

and if  $y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1$  then

$$(1.12) \quad b = y_i - \mathbf{w}^T \mathbf{x}_i = y_i (1 - \mathbf{e}_i^T H \alpha),$$

where  $\mathbf{e}_i$  is the  $i$ th column of the  $m \times m$  identity matrix  $I_m$ . Given an unseen point  $\mathbf{x}$ , the SVM decision rule for predicting its class is

$$\begin{aligned} h_{\text{SVM}}(\mathbf{x}) &= \text{sgn}(\mathbf{w}^T \mathbf{x} + b) \\ &= \text{sgn}\left(\sum_{i=1}^m y_i \alpha_i \mathbf{x}_i^T \mathbf{x} + b\right). \end{aligned}$$

At optimality, the values of  $\alpha_i$  partition the set of training points

$$S = \{\mathbf{x}_1, \dots, \mathbf{x}_m\}$$

into three parts  $S_1$ ,  $S_2$  and  $S_3$  with the following property:

$$\begin{cases} \alpha_i = 0 & \iff \mathbf{x}_i \in S_1, \quad S_1 = \{\mathbf{x}_i \mid y_i(\mathbf{w}^T \mathbf{x}_i + b) > 1 \ (\varepsilon_i = 0)\}; \\ 0 < \alpha_i < c & \iff \mathbf{x}_i \in S_2, \quad S_2 = \{\mathbf{x}_i \mid y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1 \ (\varepsilon_i = 0)\}; \\ \alpha_i = c & \iff \mathbf{x}_i \in S_3, \quad S_3 = \{\mathbf{x}_i \mid y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1 - \varepsilon_i \ (\varepsilon_i > 0)\}. \end{cases}$$

$S_2 \cup S_3$  contain the Support Vectors (SVs). It follows that at optimality the equality  $y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1 - \varepsilon_i$  holds only when  $\mathbf{x}_i$  is a SV. In other words, only the SVs determine the pair of SVM separating hyperplanes.

An important property of SVMs is that  $\alpha$  is generally sparse because usually we have  $m > n$ . The intuition explaining the sparsity is that in  $\mathbb{R}^n$  to determine a hyperplane with  $n+1$  unknowns we only need  $n+1$  equations. This is the property that people exploit to develop a fast algorithm.

## 2. Solving the SVM Dual Quadratic Programming Problem

The more examples SVMs learn from, the better predicting performance they tend to have. However, when the training set  $S_T = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_m, y_m)\}$  is large, it becomes expensive to store and access the  $m \times m$  Hessian matrix  $H$ . Hence, it becomes hard to solve the dual QP problem for standard QP solvers that take  $H$  explicitly as an input. For instance, when  $m = 10,000$ , it needs almost 1 gigabyte to store  $H$ . Many modern computers may not even have that much memory. Another concern is that when  $m > n$ ,  $H$  is rank deficient, which may bring numerical difficulties for certain solvers where the positive definiteness of  $H$  is assumed.

In this section, we describe a fast, memory efficient and easy-to-implement algorithm for solving the SVM dual QP problem. This projected Conjugate Gradient algorithm is presented in the language of linear algebra. One of the advantages of building our algorithm on basic linear algebra (BLA) operations is that these operations have been well implemented across computer platforms, so it is easy to implement this algorithm by leveraging off previous work. Also addressed are issues related to performance improvement such as how to make this algorithm adaptive to each training problem and each computer's memory architecture. Although programmed in a high-level language, as you will see in the next section, the performance of our MATLAB implementation of this algorithm is competitive with that of benchmark C/C++ codes such as SVM<sup>*light*</sup> [10] and SvmFu [20].

### 2.1. A Projected Conjugate Gradient Algorithm.

An important property of the dual QP problem is that its solution is sparse. Ignoring the training examples  $(\mathbf{x}_i, y_i)$  with  $\alpha_i = 0$  does not change the solution. For example, to compute the pair of maximal-margin hyperplanes shown in Figure 2, solving the right-hand-side separation problem based on the SVs is equivalent to solving the left-hand-side one, but it is much cheaper. Knowing which training points are the SVs in advance enables us to solve a much smaller problem. One of the main themes of this section is to exploit the sparsity property so as to solve the dual QP problem fast and with less memory usage.

To illustrate a better way to compute the maximal-margin hyperplanes, as an example, let us solve the SVM separation problem indicated in Figure 2 in the following way. Initially, we randomly choose a small set, say  $G = \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$  as our guess of the SVs. Then we solve the  $3 \times 3$  separation problem based on the working set  $G$ , and we use its solution (indicated by the dotted lines) to test the training points that are not in  $G$ . For any  $\mathbf{x}_i \in \bar{G}$  that is separated correctly by the current solution, it is still considered to be a non-SV; otherwise, we add it to  $G$ . For this problem,  $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_6\}$  are added to  $G$ . Finally, solving the updated  $6 \times 6$  separation problem gives us the optimal solution with the SVs  $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$ . Since solving these two smaller subproblems is cheaper than solving the original separation problem as a whole, this idea works. By controlling how many points to add to the working set  $G$  at each time, we can control the size of each subproblem. For instance, we can add only one point to  $G$  to formulate the second subproblem.

The above idea is quite natural from the learning perspective. We first build our knowledge of the difference between two classes based on a small number of examples. We then improve it by learning from future mistakes in distinguishing these two classes. In summary, the procedure is

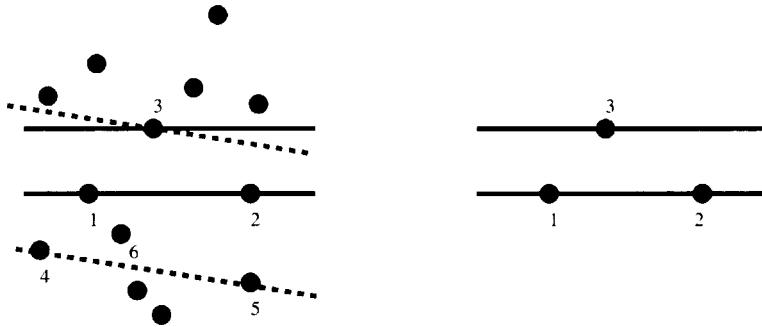


FIGURE 2. An example of a two-dimensional maximal-margin SVM separation problem. The optimal solution is the pair of lines that separate the positive (gray) and negative (dark) points with the largest gap. The SVs  $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$  and the optimal solution are indicated in the right-hand-side figure. The two dotted lines are the maximal-margin solution to separate points  $\mathbf{x}_3$  from  $\mathbf{x}_4$  and  $\mathbf{x}_5$ .

- (1) Initially, set  $\boldsymbol{\alpha} = \mathbf{0}$  and a guess of the SVs  $G$  is chosen.
- (2) The subproblem based on the working set  $G$  is constructed and solved.
- (3) If the solution from Step 2 separates all the points in  $\bar{G}$  correctly. **stop**: otherwise, a certain number of points in  $\bar{G}$  with the largest separation errors are added to  $G$ , and at the same time, the points in  $G$  that are not SVs for the current solution can be dropped to  $\bar{G}$ . then go to step 2.

When  $\boldsymbol{\alpha}$  is sparse, solving the sequence of smaller subproblems determined by the above procedure is much cheaper than solving the dual QP problem directly.

As indicated by Figure 2, this strategy can be easily appreciated from the primal problem point of view. Our goal is to show how to use this strategy to solve the dual QP problem efficiently, where some training examples may not be separated correctly by the final solution. In this section, issues such as how to construct, update, and solve the subproblems are addressed.

Other methods can be found in [17] [11] [12] [19]. The basic idea underlying all these methods is the same, that is, to decompose the original large problem into a sequence of smaller and easy-to-solve subproblems. But they differ in their decomposition strategies and their solutions to the subproblems. Some of our ideas are inspired by the Constrained Conjugate Gradient Algorithm described in [12], while [11] and [19] describe respectively the ideas underlying SVM<sup>*light*</sup> and more or less the ideas underlying SvmFu. To our knowledge, SVM<sup>*light*</sup> is currently the most popular training code.

### 2.1.1. Constructing a Subproblem.

First, let us consider in general how to project a linearly constrained QP problem onto an affine space. Suppose a new constraint is added to the dual QP problem requiring that the solution must lie in a subspace, say the column space of a matrix  $P \in \mathbb{R}^{n \times p}$ , where  $p \leq n$ . Under this restriction,  $\boldsymbol{\alpha}$  can be expressed as  $\boldsymbol{\alpha} = P\hat{\boldsymbol{\alpha}}$ , where  $\hat{\boldsymbol{\alpha}} \in \mathbb{R}^p$ . More generally,  $\boldsymbol{\alpha}$  can be written as  $\boldsymbol{\alpha} = \boldsymbol{\alpha}_0 + P\hat{\boldsymbol{\alpha}}$ , where  $\boldsymbol{\alpha}_0$  is the initial feasible value of  $\boldsymbol{\alpha}$ . For this case,  $\boldsymbol{\alpha}$  is restricted in the affine space determined by  $\boldsymbol{\alpha}_0$  and  $P$ . By substituting  $\boldsymbol{\alpha}$  with  $\boldsymbol{\alpha}_0 + P\hat{\boldsymbol{\alpha}}$  in the dual QP

problem, we have

$$(2.1) \quad \underset{\hat{\alpha}}{\text{maximize}} \quad \hat{F}(\hat{\alpha}) \equiv \hat{\alpha}^T P^T r_0 - \frac{1}{2} \hat{\alpha}^T P^T H P \hat{\alpha}$$

subject to

$$(2.2) \quad \mathbf{y}^T P \hat{\alpha} = 0,$$

$$(2.3) \quad \mathbf{0} \leq \alpha_0 + P \hat{\alpha} \leq \mathbf{c},$$

where  $r_0$  is the gradient of  $F(\alpha)$  at  $\alpha_0$ . Note that  $\mathbf{y}^T \alpha_0 = 0$  because  $\alpha_0$  is feasible. So far,  $P$  is still a general matrix. It will be specified later either directly or indirectly by its orthogonal complement  $Q$ , where  $P^T Q = 0$ .

Consider  $\alpha_0$  as an intermediate solution of the dual QP problem and  $P \hat{\alpha}$  as an update. To compute an update, we choose  $P = E_p$ , where the columns of  $E_p$  are all the  $e_i$  satisfying  $0 < e_i^T \alpha_0 < c$  (the constraint  $0 \leq \alpha_i \leq c$  is not active at  $\alpha_0$ ). If  $e_i^T \alpha_0 = 0$  or  $e_i^T \alpha_0 = c$  (the constraint  $0 \leq \alpha_i \leq c$  is active at  $\alpha_0$ ), then respectively  $e_i$  or  $-e_i$  becomes one column of  $Q$ . Note that  $P$  and  $Q$  are simply the matrix representations of the indices of the points in  $G$  and  $\bar{G}$ , or equivalently the indices of the inactive and active constraints at  $\alpha_0$ . We extend the definition of  $\bar{G}$  to incorporate the points  $x_i$ , where  $e_i^T \alpha_0 = c$ . That is, if  $x_i \in \bar{G}$  then  $\alpha_0(i)$  can be either 0 or  $c$ . Define  $\hat{H} = P^T H P$ ,  $\hat{r}_0 = P^T r_0$ ,  $\hat{\alpha}_0 = P^T \alpha_0$ , and so on so forth. With the above choice of  $P$ ,  $\hat{\alpha}$  is determined by Problem (2.1) subject to (2.2) and (2.3), which is the dual QP problem projected onto the affine subspace determined by  $\alpha_0$  and  $E_p$ :

$$(2.4) \quad \underset{\hat{\alpha}}{\text{maximize}} \quad \hat{F}(\hat{\alpha}) \equiv \hat{\alpha}^T \hat{r}_0 - \frac{1}{2} \hat{\alpha}^T \hat{H} \hat{\alpha}$$

subject to

$$(2.5) \quad \hat{\mathbf{y}}^T \hat{\alpha} = 0,$$

$$(2.6) \quad \mathbf{0} \leq \hat{\alpha}_0 + \hat{\alpha} \leq \mathbf{c}.$$

Note that  $\mathbf{0}$  and  $\mathbf{c}$  are of size  $p$ . Since  $P \hat{\alpha}$  is orthogonal to the column space of  $Q$ , it does not affect any active constraints at  $\alpha_0$ . In other words, the update determined by the above QP problem only improves the part of  $\alpha_0$  whose values are strictly between 0 and  $c$ . When  $p$  is small, it is cheap to solve the above problem.

To deal with the equality constraint (2.5), Problem (2.4) is projected again onto the hyperplane determined by this constraint. We choose  $Q = \hat{\mathbf{y}}$  and  $P = I - \frac{\hat{\mathbf{y}} \hat{\mathbf{y}}^T}{p}$  for this projection. It is easy to see that the column spaces of  $P$  and  $Q$  are orthogonal complements to each other, and that  $I - P$  is the orthogonal projector onto the column space of  $Q$ . To reuse the notation  $\hat{\alpha}$ , in (2.4), (2.5) and (2.6),  $\hat{\alpha}$  is replaced with  $P \hat{\alpha}$ , where  $P = I - \frac{\hat{\mathbf{y}} \hat{\mathbf{y}}^T}{p}$ . The resulting subproblem has a simpler form:

$$(2.7) \quad \underset{\hat{\alpha}}{\text{maximize}} \quad \hat{F}(\hat{\alpha}) \equiv \hat{\alpha}^T P^T \hat{r}_0 - \frac{1}{2} \hat{\alpha}^T P^T \hat{H} P \hat{\alpha}$$

subject to

$$(2.8) \quad \mathbf{0} \leq \hat{\alpha}_0 + P \hat{\alpha} \leq \mathbf{c}.$$

If  $\hat{\alpha}$  is the solution to the above problem, then the update is

$$E_p(I - \frac{\hat{y}\hat{y}^T}{p})\hat{\alpha}.$$

Remember that in Problem (2.4) subject to (2.5) and (2.6),  $P = E_p$  and  $E_p\hat{\alpha}$  gives the update (the notation  $\hat{\alpha}$  is reused in Problem (2.7) subject to (2.8)).

### 2.1.2. Solving the Subproblem by the Conjugate Gradient Method.

If the box constraint (2.8) turns out to be loose, i.e., not active, then the above problem is equivalent to a linear system

$$P^T \hat{H} P \hat{\alpha} = P^T \hat{r}_0,$$

or equivalently

$$(2.9) \quad P \hat{H} P \hat{\alpha} = P \hat{r}_0 \quad (P \text{ is symmetric}).$$

Since  $P \hat{H} P$  is symmetric and usually positive definite under the above assumption, this linear system could be solved numerically by the Conjugate Gradient (CG) method [8] [22]:

$$\begin{aligned} \hat{\alpha}^{(0)} &= 0, \quad \gamma^{(0)} = P \hat{r}_0, \quad \rho^{(0)} = \gamma^{(0)} \\ \lambda &= \frac{\gamma^{(l-1)T} \gamma^{(l-1)}}{\rho^{(l-1)T} P \hat{H} P \rho^{(l-1)}} \\ \hat{\alpha}^{(l)} &= \hat{\alpha}^{(l-1)} + \lambda \rho^{(l-1)} \\ \gamma^{(l)} &= \gamma^{(l-1)} - \lambda P \hat{H} P \rho^{(l-1)} \\ \mu &= \frac{\gamma^{(l)T} \gamma^{(l)}}{\gamma^{(l-1)T} \gamma^{(l-1)}} \\ \rho^{(l)} &= \gamma^{(l)} + \mu \rho^{(l-1)}. \end{aligned}$$

From the fact  $P^2 = P$ , it follows that the above CG iterations can be simplified as

$$\begin{aligned} \hat{\alpha}^{(0)} &= 0, \quad \gamma^{(0)} = P \hat{r}_0, \quad \rho^{(0)} = \gamma^{(0)} \\ \lambda &= \frac{\gamma^{(l-1)T} \gamma^{(l-1)}}{\rho^{(l-1)T} \hat{H} \rho^{(l-1)}} \\ \hat{\alpha}^{(l)} &= \hat{\alpha}^{(l-1)} + \lambda \rho^{(l-1)} \\ \gamma^{(l)} &= \gamma^{(l-1)} - \lambda P(\hat{H} \rho^{(l-1)}) \\ \mu &= \frac{\gamma^{(l)T} \gamma^{(l)}}{\gamma^{(l-1)T} \gamma^{(l-1)}} \\ \rho^{(l)} &= \gamma^{(l)} + \mu \rho^{(l-1)}. \end{aligned}$$

Note that  $\hat{H}$  is a  $p \times p$  matrix. Since  $P = I - \frac{\hat{y}\hat{y}^T}{p}$ , the multiplication of  $P$  with a vector is cheap. It only involves two Level-1 (vector-vector) operations. The most expensive operation above is the Level-2 (matrix-vector) operation  $\hat{H} \rho^{(l-1)}$ . When  $p$  is small, it is cheap to compute the above operations and the storage requirement (one  $p \times p$  matrix and four  $p$ -vectors) is also small.

However, to solve Problem (2.7) subject to (2.8) with the above algorithm, the box constraint (2.8) has to be considered. If  $0 \leq (\hat{\alpha}_0)_i + (P\hat{\alpha})_i \leq c$  becomes active during the Conjugate Gradient iterations, then its global index representation  $e_{l_i}$  is added to  $Q$ . At the same time, some old active constraints can be relaxed by moving their index representations from  $Q$  to  $P$ , if doing so improves the objective function. After  $P$  and  $Q$  is updated, the subproblem (2.7) subject to (2.8) is reformulated correspondingly and solved again. In the following section, the stopping criteria and the criteria for relaxing active constraints are given. An important fact worth to point out is that  $\hat{H}$  is not guaranteed to be in full rank. Therefore, we need to control the steps of the CG iterations to keep the solution from blowing up.

### 2.1.3. The Optimality Conditions.

Note that Problem (2.7) subject to (2.8) is the compact version of Problem (2.1) subject to (2.2) and (2.3) with

$$\begin{aligned} Q &= [\pm e_{i_1}, \pm e_{i_2}, \dots, \pm e_{i_q}, y] \\ &= [E_q, y] \end{aligned}$$

and

$$P = I - Q(Q^T Q)^{-1} Q^T,$$

where  $E_q$  is the orthogonal complement of  $E_p$  ( $m = p + q$ ), and the column space of  $P$  is the orthogonal complement of the column space of  $Q$ . In other words, the previous two projections are incorporated in the current definition of  $P$ . Let us define

$$d = Pr$$

and

$$\beta = -[I_{q \times q}, 0](Q^T Q)^{-1} Q^T r,$$

where  $r = \mathbf{1} - H\alpha$  and  $P$  is a  $n \times n$  matrix. The optimality conditions [1] [3] tell that  $\alpha$  is optimal iff

$$(2.10) \quad d = \mathbf{0} \text{ and } \beta \geq \mathbf{0}.$$

Note that  $d$  is the projected gradient onto the search space spanned by  $P$ . If  $d = \mathbf{0}$ , then we have reached a stationary point. The condition  $\beta \geq \mathbf{0}$  implies that relaxing the active constraints represented by  $Q$  does not improve the objective function nearby. It follows that if (2.10) is true then  $\alpha$  is a local optimum, thus a global optimum due to the convex property of a linearly constrained QP problem.

Since  $(Q^T Q)^{-1}$  has a simple closed form:

$$\left[ \begin{array}{cc} I + \frac{E_q^T y y^T E_q}{p} & \frac{-E_q^T y}{p} \\ \frac{-y^T E_q}{p} & \frac{1}{p} \end{array} \right],$$

it is easy to compute  $d$  and  $\beta$ . For instance,

$$(2.11) \quad \beta = E_q^T r + \frac{E_q^T y ((E_q^T y) E_q^T r - y^T r)}{p},$$

where  $E_q$  is an indexing operator. Each time when active constraints are to be relaxed, we start with the one that has the most negative  $\beta_i$ .

### 2.1.4. The Algorithm.

As a summary of the previous discussion, the algorithm is outlined briefly as the following:

```

 $\alpha = \mathbf{0};$ 
initialize  $E_p$ ;
while  $d \neq \mathbf{0}$  or  $\beta < 0$ 
    at most  $k$  steps of the CG iterations for Problem (2.7) subject to (2.8):
    update  $\alpha$  and  $r$ :
    
$$\alpha = \alpha + E_p(I - \frac{\hat{y}\hat{y}^T}{p})\hat{\alpha};$$

    
$$r = r - HE_p(I - \frac{\hat{y}\hat{y}^T}{p})\hat{\alpha};$$

    compute  $\beta$ :
    relax at most  $l$  active constraints with the most negative  $\beta_i$ :
    update  $E_p$ ;
    compute the columns of  $H$  corresponding to the relaxed constraints:
    update  $HE_p$  and  $\hat{H}$ ;
end

```

During each iteration of this algorithm, only  $p$  columns of  $H$  ( $HE_p$ ) are used. Thus, there is no need to precompute  $H$  so that a lot of flops (floating point operations) and memory can be saved. In our implementation of this algorithm, two matrices  $X = [\mathbf{x}_1, \dots, \mathbf{x}_m]$  and  $HE_p$  are stored in memory. Since  $n$  and  $p$  are generally much smaller than  $m$ , most modern computers can meet the memory requirement for problems of size  $m = O(10,000)$ . Later in this paper, we will show an alternative which consumes less memory by only using part of  $HE_p$ , but taking more steps to converge.

Note that the  $i$ th column of  $H$  is computed only when  $\alpha_i = 0$  or  $\alpha_i = c$  is relaxed. It turns out that computing the columns of  $H$  is the most expensive operation in this algorithm. When  $l$  is large, it can be considered as a Level-3 (matrix-matrix multiplication) operation. Although Level-1 and Level-2 operations are cheaper in terms of flops, Level-3 operations can be implemented more efficiently by exploiting the computer memory hierarchy. More discussions will be given on the implementation of these operations in the next section.

The setting of parameters  $k$  and  $l$  affects the performance. We will discuss more on how to choose  $k$  and  $l$  in the next section. In our implementation,  $k$  and  $l$  are determined adaptively based on each training set. To initialize  $G$ , we can randomly choose a small number of training points from the training set. An alternative is to compute an initial guess of the pair of separating hyperplanes and choose the points near the two hyperplanes as the candidates for  $G$ .

Note that the residual or the gradient  $e_i^T r = 1 - y_i \mathbf{w}^T \mathbf{x}_i$  tells how well the current solution separates  $\mathbf{x}_i$ . When (2.10) is true, the optimality is achieved. As we have seen, this algorithm successfully exploits the sparsity property of SVMs. When the final solution is sparse, it solves the dual QP problem with much less flops and memory consumption. Meanwhile, this algorithm can be easily implemented

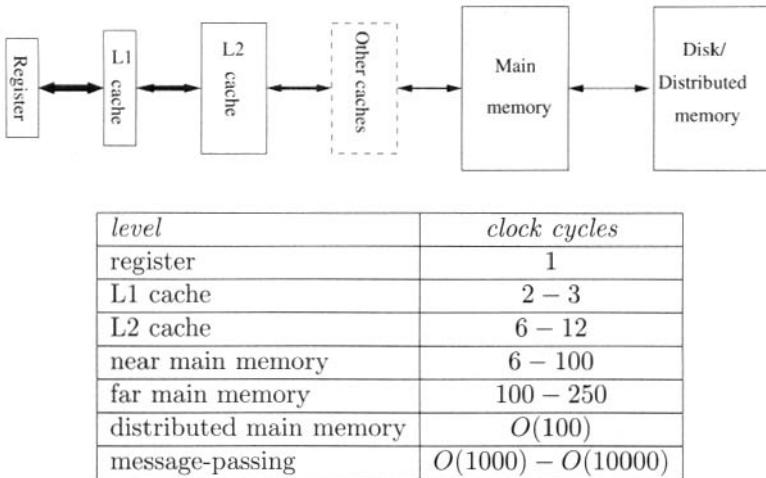


FIGURE 3. A model of the modern computer memory hierarchy.  
The statistics in the table above are cited from [7].

by MATLAB. For details of our MATLAB implementation, please refer to the downloadable codes [25].

If each subproblem is solved exactly, then the objective function is improved strictly each time. However, from the performance point of view, it is better off to solve each subproblem approximately except the last one. Issues related to further speeding up the convergence of this algorithm are addressed in the following section.

## 2.2. Speed Considerations.

The sparsity property of SVMs enables us to avoid unnecessary memory consumption and flops. To achieve better performance, in this section we discuss how to make this algorithm adaptive to each computer’s memory hierarchy and each training problem.

### 2.2.1. Being Adaptive to the Computer Memory Hierarchy.

When large data sets are involved in computations, the number of flops is not an accurate indicator of the running time. The cost of accessing data must also be taken into account. In Figure 3, a model of modern computer memory hierarchy is given. Loading data from main memory to caches and writing data from caches back to main memory are expensive operations. To reduce the cost of accessing data, computations should be arranged so that more data when needed can be found at the top (left) levels of the hierarchy. Decomposing a large problem into smaller subproblems is a good strategy. It is well known that Level-3 operations can be implemented more efficiently than Level-1 and Level-2 operations because of the above model.

In general, fast codes are developed by explicitly using the information of each configuration of the above model. To achieve portable performance, we want our implementation to be adaptive to each machine’s memory hierarchy as much as possible. Since the basic operations in our algorithm are BLA operations, it is important to implement them adaptively. To achieve this goal, we use ATLAS BLAS [26] [27] [28], a software package for BLA operations that is automatically tuned for each machine. Thanks to MATLAB’s inclusion of ATLAS BLAS starting

from its version 6. Better performance and portability can be achieved now from MATLAB.

### 2.2.2. Setting $k$ and $l$ Adaptively.

As mentioned in the previous section, the running time of this algorithm depends on the setting of  $k$  and  $l$ . In this section, we discuss how to set these two parameters adaptively for each training problem.

The convergence rate of this algorithm is related to the spectrum of  $H$  and the value of  $c$ . Generally, the leading eigenvalues of  $H$  are well separated. The number of these leading eigenvalues is a good indicator of the size of  $S_1$ . While the value of  $c$  affects the size of  $S_2$ . The smaller  $c$  is, the larger  $S_2$ . For the extreme case when the training set is separable and  $c$  is set very large (such that the constraints  $\alpha_i \leq c$  never become active), the spectrum of  $H$  is an important factor determining the running time. Based on this observation, we set  $k$  and  $l$  adaptively using the information of the spectrum of  $H$ .

From the convergence property of the CG method, we know that when using the CG method to solve a symmetric positive definite linear system with  $j$  well-separated leading eigenvalues, the residual tends to be reduced quickly during the first  $j$  iterations. According to this property, we estimate the number of leading eigenvalues by  $\hat{k}$ , the number of iterations for the CG method to reduce the residual norm of the first subproblem to a scale of 0.01.

We use  $k$  to control the steps of the CG iterations, because there is no need to compute Problem (2.7) subject to (2.8) exactly if it is not the final subproblem. Another reason to not solve every subproblem exactly is due to the rank deficiency of  $H$ . The CG method converges slowly or may not even converge for ill-conditioned matrices.  $k$  should be bounded by the rank of  $H$ . For example, if  $\mathbf{x}_i \in \mathbb{R}^2$  then  $\hat{H}$  has two nonzero eigenvalues in general. It makes sense to solve the subproblems with only two steps of the CG iterations before the solution blows up. In our algorithm,  $k$  is set to be  $\hat{k}$  obtained from the initial subproblem.

When  $l$  active constraints are relaxed,  $[\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_l}]^T X$  is computed to generate the corresponding columns of  $H$  ( $H$  is not precomputed). Considering that  $[\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_l}]^T X$  is a Level-3 operation when  $l$  is large, we tend to relax all the active constraints with negative  $\beta_i$  (because Level-3 operations can be implemented efficiently). However, with this strategy we increase the probability that some previously relaxed constraints become active again, which may slow down the convergence. Our observation is that when the number of the SVs in  $S_1$  is small, it is better off to set  $l$  small. Since the number of  $H$ 's leading eigenvalues provides information about the size of  $S_1$ , in our algorithm  $l$  is also set to be  $\hat{k}$ .

## 3. Numerical Experiments

The MATLAB implementation of this algorithm is named “FMSvm”, where “FM” stands for “Fast MATLAB”. In this section, FMSvm [25] is compared with two benchmark C and C++ codes, SVM<sup>light</sup> and SvmFu respectively.

We use two training sets Digit17 and Face to compare the convergence rates of these three training codes. Digit17 is a sparse training set containing 13007 samples of handwritten digits 1 and 7, which are obtained from the MNIST handwritten-digit database. This database has become a standard for testing and comparing different learning algorithms. The other training set is a face detection training set obtained from MIT AI Lab, which is dense and contains 31022 examples. The face

Training sets: Digit17-6000 and Digit17							
l(1)	l(0.1)	p2(0.1)	p2(0.001)	r10(10)	r10(1)	r3(1)	r3(0.1)
Training sets: Face-6000, Face-13901 and Face							
l(4)	l(1)	l(0.1)	p2(1)	p2(0.001)	r10(10)	r3(10)	r3(1)

TABLE 1. Choices of kernels and  $c$ . Here, “l” stands for the linear kernel, “p2” stands for the polynomial kernel with degree 2 and “rx” stands for the radial basis function kernel with  $\sigma = x$ . The number in the parenthesis is the value of  $c$ . The definitions of these kernel functions are listed in the appendix.

detection problem is interesting because it represents a class of similar problems, such as how to detect cancers in medical images and how to detect enemy tanks in satellite images. Digit17 is easy to separate and it is balanced (6742 positive examples vs. 6265 negative examples). While Face is harder to separate and unbalanced (2901 positive examples vs. 28121 negative examples). The training points in Digit17 and Face are of dimensions 784 and 361 respectively. The two training set are downloadable at [25].

Both the SVM<sup>light</sup> and SvmFu training functions have parameters whose setting affects their running time<sup>1</sup>. Bad choices can make the convergence very slow. Unfortunately, the optimal setting of these parameters are not known a priori. If they are not set, the default setting is taken. The FMSvm training function also has two parameters  $k$  and  $l$ , but users are not required to set them. Instead, it tries to estimate the optimal setting by itself based on each particular training problem. In this section we compare FMSvm with SVM<sup>light</sup> and SvmFu using both default and estimated optimal settings.

From Digit17 and Face, we create five training sets: Digit17-6000, Digit17, Face-6000, Face-13902 and Face, with sizes 6000, 13007, 6000, 13902 and 31022 respectively. The two smallest training sets are used to obtain an estimate of the optimal settings for SVM<sup>light</sup> and SvmFu. The estimated optimal setting for each training function is the setting among ten trials which gives the best performance. Then on larger training sets Digit17, Face-13902 and Face, FMSvm is compared against SVM<sup>light</sup> and SvmFu with the estimated optimal settings. For each training set, different kernel functions and choices of  $c$  as listed in Table 1 are used for the comparison.

In Figure 4, we can see that for the easy-to-separate training set Digit17-6000, there is no big difference in performance between FMSvm and the other two with default settings. But for Face-6000’s case l(4), FMSvm does much better than both SVM<sup>light</sup> and SvmFu. The timing results of FMSvm against SVM<sup>light</sup> and SvmFu plotted in the right column of Figure 4 are for the case where the estimated optimal settings are used for all of them. The estimated optimal settings obtained here for SVM<sup>light</sup> and SvmFu are used for later comparisons shown in Figure 5 and Figure 6. These estimated optimal settings are listed in the appendix.

<sup>1</sup>The convergence time is measured by cpu seconds. For SVM<sup>light</sup> and SvmFu, the time spent in loading the input data file is excluded. For instance, SvmFu needs around 24 cpu seconds to load the training set Face, while SVM<sup>light</sup> needs around 115 cpu seconds. All the timing results in this paper are obtained on the sever `newton.mit.edu`, which is a linux machine with 2 1.2 GHZ Athlon MP processors and 2 GB memory.

## Using Default Settings

## Using Estimated Optimal Settings

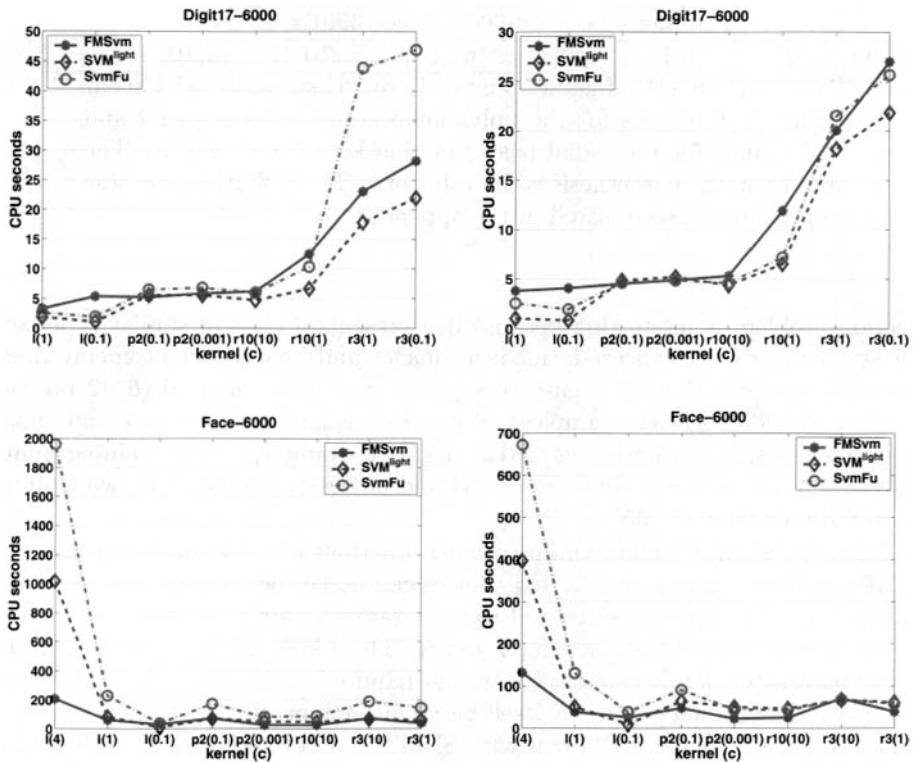


FIGURE 4. Timing results on the two training subsets of size 6000. Plots in the left column are the timing results of FMSvm against  $\text{SVM}^{\text{light}}$  and SvmFu with default parameter settings. For problems of size larger than 2000, the default setting for SvmFu is “ $(h, c)$ ” =  $(2000, 0)$  and the default setting for  $\text{SVM}^{\text{light}}$  is “ $(q, n, m)$ ” =  $(10, 10, 40)$ . However, for training set Face-6000 we use  $(20, 10, 40)$  as  $\text{SVM}^{\text{light}}$ ’s default setting, following the hint to choose  $n < q$  to prevent zig-zagging, where  $(20, 10, 40)$  is the estimated optimal setting for Digit17-6000. Note that in  $\text{SVM}^{\text{light}}$  and SvmFu,  $h, c$  and  $q, n, m$  have different meanings. Plots in the right column are the timing results of FMSvm against  $\text{SVM}^{\text{light}}$  and SvmFu, where the estimated optimal settings are used for all of them. With default settings, for the easy-to-separate training set Digit17-6000, there is no big difference in performance between the FMSvm training function and the other two. But for Face-6000 ’s case I(4), FMSvm does much better than both  $\text{SVM}^{\text{light}}$  and SvmFu.

### Comparison of FMSvm with SVM<sup>*light*</sup> and SvmFu with the estimated optimal parameter settings

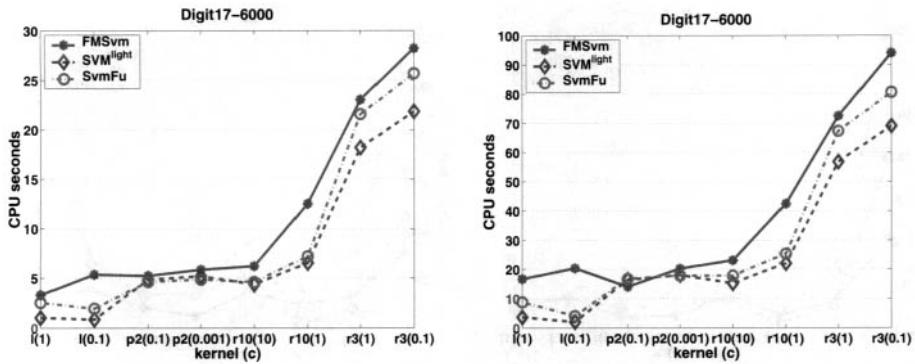


FIGURE 5. FMSvm VS. SVM<sup>*light*</sup> and SvmFu. The estimated optimal parameter settings are used for SVM<sup>*light*</sup> and SvmFu. For those parameters that seem to depend on the size of the training set, their values are adjusted proportionally as the size increases. All the settings are listed in the appendix. We can see that FMSvm (with parameters set automatically) matches the estimated best performance of SVM<sup>*light*</sup> and SvmFu on the training sets Digit17-6000 and Digit17.

Since the optimal parameter settings for all the three training functions are not known a priori, the default setting or a setting from an arbitrary guess may cause very slow convergence. FMSvm tries to avoid this situation by being adaptive to each training problem. Shown in Figure 5 and Figure 6 are the timing results of FMSvm against SVM<sup>*light*</sup> and SvmFu, where the estimated optimal settings are used for SVM<sup>*light*</sup> and SvmFu. We can see that FMSvm matches the estimated best performance of SVM<sup>*light*</sup> and SvmFu on the two training sets Digit17 and Face.

If we measure the difference between a setting  $A$  and the estimated optimal setting  $O$  by the ratio of relative slowdown:

$$s = \frac{\text{time}(A) - \text{time}(O)}{\text{time}(O)},$$

then in Figure 4, for each training function we get 16 such ratios that measure the difference between the setting used in the left column and the estimated optimal setting used in the right column. The mean and standard deviation of these ratios are respectively  $(0.16, 0.14)$ ,  $(0.24, 0.43)$  and  $(0.76, 0.60)$  for FMSvm, SVM<sup>*light*</sup> and SvmFu. We can see that the parameter setting used by FMSvm is closer to the corresponding estimated optimal.

In summary, FMSvm's idea of being adaptive to each training problem works, which helps preventing the slow convergence caused by bad parameter settings. Based on training sets Digit17 and Face, we can see that FMSvm matches the best performance of SVM<sup>*light*</sup> and SvmFu, while having the features provided by a high-level programming language.

### Comparison of FMSvm with $\text{SVM}^{\text{light}}$ and SvmFu with the estimated optimal parameter settings

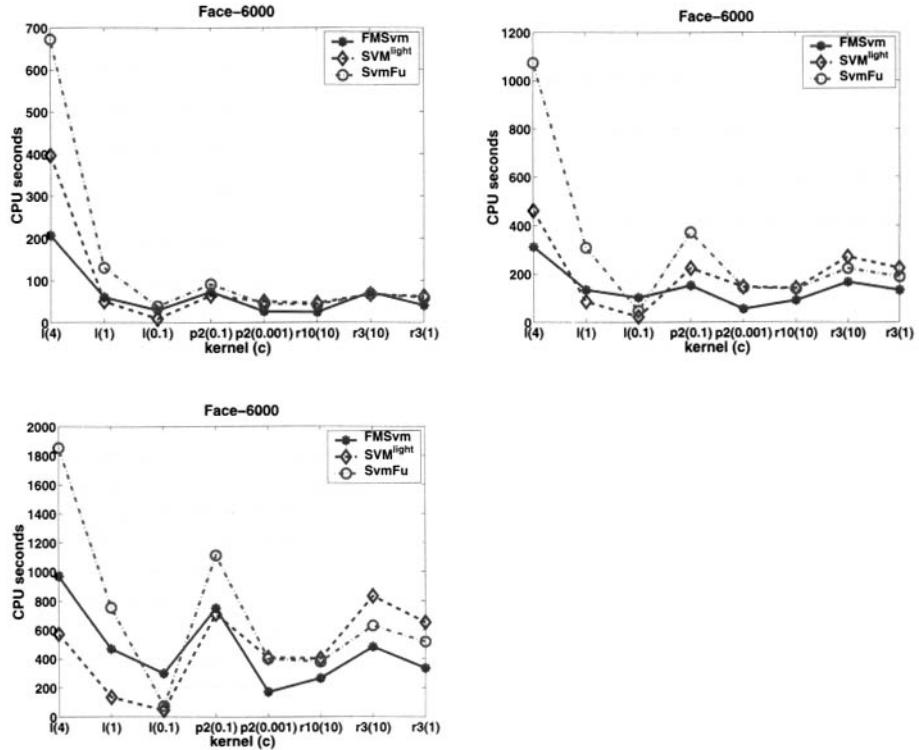


FIGURE 6. FMSvm VS.  $\text{SVM}^{\text{light}}$  and SvmFu. The estimated optimal parameter settings are used for the training functions of  $\text{SVM}^{\text{light}}$  and SvmFu. For those parameters that seem to depend on the size of the training set, their values are adjusted proportionally as the size increases. All the settings are listed in the appendix. We can see that FMSvm (with parameters set automatically) matches the estimated best performance of  $\text{SVM}^{\text{light}}$  and SvmFu on the training sets Face-6000, Face-13901 and Face.

#### 4. Distributing Large Training Problems

The basic memory requirement for our algorithm is to store the training example matrix  $X$  and the computed part of the Hessian matrix  $HE_p$ . Instead of  $E_p^T HE_p$ ,  $HE_p$  is stored because we need to update the residual

$$r = r_0 - HE_p \left( I - \frac{\hat{y}\hat{y}^T}{p} \right) \hat{\alpha}.$$

To control memory consumption, we can update part of the residual so that only the corresponding rows of  $HE_p$  are needed. From the primal problem point of view, this is equivalent to testing part of the training points in  $\overline{G}$  using the current solution. This approach can be easily extended to distribute large training problems. The

idea is to use the master process to compute the subproblems, where only  $E_p^T H E_p$  is needed, and let the slave processes to update the residual in parallel. Updating the residual is expensive because the corresponding part of  $H$  has to be computed.

To make our MATLAB code support distributed computations, we use an extended MATLAB environment called MATLAB\*P [9], where an almost transparent interface to distributed matrices and the operations on them is provided. MATLAB\*P consists of two parts, a MATLAB front end and a server living in a supercomputer. Distributed matrices and the operations on them are stored and executed respectively on the server. Unlike MATLAB's transparent support to sparse matrices, small modification is required to make ordinary MATLAB codes to support MATLAB\*P's distributed matrices. But once MATLAB\*P knows that  $X$  is a distributed matrix, all the operations involving  $X$  are automatically executed by the server. We expect that extending FMSvm to support MATLAB\*P's distributed matrices should not be a difficult task.

## 5. Conclusion and Future work

While successfully exploiting the sparse property of SVMs, this algorithm also provides the capability to be adaptive to the training problem and the computer memory hierarchy. The resulting implementation is not only efficient but also highly portable and easy-to-use. We expect that further performance improvement can be achieved using preconditioning techniques. This could be a feature for the next version of FMSvm.

A rigorous analysis of the convergence behavior of this algorithm is subject to future study. The techniques and heuristics employed here to speed up the convergence make this analysis hard. Although FMSvm works well on the training sets we have tried, so far we can not guarantee that this algorithm converges for all the cases. In our future work, we want to better understand the convergence behavior of this algorithm and to make it more adaptive. We also want to test this algorithm with more training problems and on more computer platforms.

## Appendix A. List of Kernel Functions

- Linear kernel:  $k(\mathbf{s}, \mathbf{t}) = \mathbf{s}^T \mathbf{t}$ ,  $\mathbf{s}, \mathbf{t} \in \mathbb{R}^n$ .
- Polynomial kernel:  $k(\mathbf{s}, \mathbf{t}) = (\mathbf{s}^T \mathbf{t} + b)^d$ ,  $\mathbf{s}, \mathbf{t} \in \mathbb{R}^n$  and  $b, d \in \mathbb{R}$ .
- Radial basis function kernel:  $k(\mathbf{s}, \mathbf{t}) = e^{-\|\mathbf{s}-\mathbf{t}\|^2/(2\sigma^2)}$ ,  $\mathbf{s}, \mathbf{t} \in \mathbb{R}^n$  and  $\sigma \in \mathbb{R}$ .

## Appendix B. List of estimated optimal settings for SVM<sup>light</sup> and SvmFu

- Table 2: The Estimated Optimal Settings for Training Sets Digit17-6000 and Digit17.
- Table 3: The Estimated Optimal Settings for Training Sets Face-6000, Face-13901 and Face.

## References

- [1] Mordechai Ariel. *Nonlinear Programming: Analysis and Methods*. Prentice-Hall, N.J., 1976.
- [2] D. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, 1995.
- [3] John C. G. Boot. *Quadratic Programming*. Rand McNally & Company, Chicago, 1964.

Kernels (c)	Digit17-6000		Digit17	
	SVM <sup>light</sup> 3.5	SvmFu3.0	SVM <sup>light</sup> 3.5	SvmFu3.0
l(1)	(20, 10, 40)	(2000, 100)	(20, 10, 80)	(2000, 200)
l(0.1)	(20, 10, 40)	(2000, 0)	(20, 10, 80)	(2000, 0)
p2(0.1)	(20, 10, 40)	(6000, 0)	(20, 10, 80)	(13007.0)
p2(0.001)	(20, 10, 40)	(6000, 0)	(20, 10, 80)	(13007.0)
r10(10)	(20, 10, 40)	(6000, 0)	(20, 10, 80)	(13007.0)
r10(1)	(20, 10, 40)	(6000, 0)	(20, 10, 80)	(13007.0)
r3(1)	(20, 10, 40)	(6000, 0)	(20, 10, 80)	(13007.0)
r3(0.1)	(20, 10, 40)	(6000, 0)	(20, 10, 80)	(13007.0)

TABLE 2. The Estimated Optimal Settings for Training Sets Digit17-6000 and Digit17.

Kernels (c)	Face-6000		Face-13901		Face	
	SVM <sup>light</sup> 3.5	SvmFu3.0	SVM <sup>light</sup> 3.5	SvmFu3.0	SVM <sup>light</sup> 3.5	SvmFu3.0
l(4)	(60, 20, 40)	(6000, 0)	(60, 20, 80)	(13901, 0)	(60, 20, 200)	(31022.0)
l(1)	(60, 20, 40)	(6000, 0)	(60, 20, 80)	(13901, 0)	(60, 20, 200)	(31022.0)
l(0.1)	(40, 10, 40)	(2000, 40)	(40, 10, 80)	(2000, 80)	(40, 10, 200)	(2000, 200)
p2(1)	(40, 10, 40)	(6000, 0)	(40, 10, 80)	(13901, 0)	(40, 10, 200)	(31022.0)
p2(0.001)	(60, 20, 40)	(6000, 0)	(60, 20, 80)	(13901, 0)	(60, 20, 200)	(31022.0)
r10(10)	(20, 10, 40)	(6000, 0)	(20, 10, 80)	(13901, 0)	(20, 10, 200)	(31022.0)
r3(10)	(20, 10, 40)	(6000, 0)	(20, 10, 80)	(13901, 0)	(20, 10, 200)	(31022.0)
r3(1)	(20, 10, 40)	(6000, 0)	(20, 10, 80)	(13901, 0)	(20, 10, 200)	(31022.0)

TABLE 3. The Estimated Optimal Settings for Training Sets Face-6000, Face-13901 and Face.

- [4] L. Bottou, C. Cortes, J. Denker, H. Drucker, I. Guyon, L. Jackel, Y. LeGun, U. Müller, E. Säckinger, P. Simard and V. Vapnik. Comparison of Classifier Methods: a Case Study in Handwritten Digit Recognition. *Proceedings of the 12th International Conference on Pattern Recognition and Neural Networks*. Jerusalem. 77-87. IEEE Computer Society Press, 1994.
- [5] Christopher J.C. Burges. *A Tutorial on Support Vector Machines for Pattern Recognition*. Knowledge Discovery and Data Mining. 2(2). 1998.
- [6] Nello Cristianini, John Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000.
- [7] Craig C. Douglas, Gundolf Haase, Jonathan Hu, Markus Kowarschik, Ulrich Rüde and Christian Weiss. *Portable Memory Hierarchy Techniques For PDE Solvers: Part I*. SIAM NEWS, Volume 33/Number 5. June 2000.
- [8] Gene H. Golub, Charles F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore and London, 1996.
- [9] Parry Husbands. *Interactive Supercomputing*. Ph.D Thesis, M.I.T.. 1999.
- [10] Thorsten Joachims. SVM<sup>light</sup> 3.5. <http://svmlight.joachims.org>.
- [11] Thorsten Joachims. Making Large-Scale Support Vector Machine Learning Practical. *Advances in Kernel Methods: Support Vector Learning*, edited by Bernhard Schölkopf, Christopher J.C. Burges and Alexander J. Smola. The MIT Press, Cambridge, 1998.
- [12] Linda Kaufman. Solving the Quadratic Programming Problem Arising in Support Vector Classification. *Advances in Kernel Methods: Support Vector Learning*, edited by Bernhard Schölkopf, Christopher J.C. Burges and Alexander J. Smola. The MIT Press, Cambridge, 1998.

- [13] M. Kirby, L. Sirovich. Application of the Karhunen-Loëve Procedure for the Characterization of Human Faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):103-108, 1990.
- [14] U. Krebel. The Impact of the Learning-Set Size in Handwritten Digit Recognition. *Artificial Neural Networks – ICANN'91*, edited by T. Kohonen, 1685-1689, Amsterdam, 1991. North-Holland.
- [15] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. Jackel. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1:541-551, 1989.
- [16] Y. LeCun, L. Jackel, L. Bottou, C. Cortes, J. Denker, H. Drucker, I. Guyon, U. Müller, E. Säckinger, P. Simard and V. Vapnik. Comparison of Learning Algorithms for Handwritten Digit Recognition. *Proceedings ICANN'95 – International Conference on Artificial Neural Networks*, edited by F. Fogelman-Soulie and P. Gallinari. Volume II, 53-60, Nanterre, France, 1995.
- [17] E. Osuna, R. Freund and F. Girosi. An Improved Training Algorithm for Support Vector Machines. *Neural Networks for Signal Processing VII – Proceeding of the 1997 IEEE Workshop*, edited by J. Principe, L. Gile, N. Morgan and E. Wilson, 511-520, New York, 1997. IEEE.
- [18] E. Osuna, R. Freund and F. Girosi. Training Support Vector Machines: An Application to Face Detection. *Proceedings, Computer Vision and Pattern Recognition'97*, 130-136, 1997.
- [19] John C. Platt. Fast Training of Support Vector Machines Using Sequential Minimal Optimization. *Advances in Kernel Methods: Support Vector Learning*, edited by Bernhard Schölkopf, Christopher J.C. Burges and Alexander J. Smola, The MIT Press, Cambridge, 1998.
- [20] Ryan Rifkin. *SvmFu 3.0*. <http://fpn.mit.edu/SvmFu/>.
- [21] Edited by Bernhard Schölkopf, Christopher J.C. Burges and Alexander J. Smola. *Advances in Kernel Methods: Support Vector Learning*. The MIT Press, Cambridge, 1999.
- [22] Lloyd N. Trefethen, David Bau, III. *Numerical Linear Algebra*. SIAM, Philadelphia, 1997.
- [23] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, New York, 1995.
- [24] Vladimir N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, Inc, 1998.
- [25] Tong Wen. *FMSvm 1.0*. <http://math.mit.edu/~tonywen/FMSvm/>.
- [26] R. Whaley and J. Dongarra. Automatically Tuned Linear Algebra Software (ATLAS). *SC '89 Proceedings* (Electronic Publication). IEEE Publication.
- [27] R. Clint Whaley, Antoine Petitet, and Jack Dongarra. Automated Empirical Optimizations of Software and the ATLAS Project. *Parallel Computing*, Volume 27, Numbers 1-2, pp 3-25, 2001.
- [28] R. Clint Whaley, Antoine Petitet, Jack J. Dongarra. *Automated Empirical Optimization of Software and the ATLAS Project*. <http://math-atlas.sourceforge.net/>.

DEPARTMENT OF MATHEMATICS, MASSACHUSETTS INSTITUTE OF TECHNOLOGY, CAMBRIDGE,  
MA 02139

*E-mail address:* `tonyweng@math.mit.edu`

DEPARTMENT OF MATHEMATICS & LABORATORY OF COMPUTER SCIENCE, MASSACHUSETTS  
INSTITUTE OF TECHNOLOGY, CAMBRIDGE, MA 02139

*E-mail address:* `edelman@math.mit.edu`

U.S. ARMY TANK-AUTOMOTIVE & ARMAMENTS COMMAND, AUTOMOTIVE RESEARCH CENTER,  
WARREN, MI 48397

*E-mail address:* `garsichd@tacom.army.mil`

*This page intentionally left blank*

# A Displacement Approach to Decoding Algebraic Codes

V. Olshevsky and M. Amin Shokrollahi

**ABSTRACT.** Algebraic coding theory is one of the areas that routinely gives rise to computational problems involving various structured matrices, such as Hankel, Vandermonde, Cauchy matrices, and certain generalizations thereof. Their structure has often been used to derive efficient algorithms; however, the use of the structure was pattern-specific, without applying a unified technique. In contrast, in several other areas, where structured matrices are also widely encountered, the concept of displacement rank was found to be useful to derive efficient algorithms in a unified manner (i.e., not depending on a particular pattern of structure). The latter technique allows one to “compress,” in a unified way, different types of  $n \times n$  structured matrices to only  $\alpha n$  parameters. This typically leads to computational savings (in many applications the number  $\alpha$ , called the displacement rank, is a small fixed constant).

In this paper we demonstrate the power of the displacement structure approach by deriving in a unified way efficient algorithms for a number of decoding problems. We accelerate the Sudan’s list decoding algorithm for Reed-Solomon codes, its generalization to algebraic-geometric codes by Shokrollahi and Wasserman, and the improvement of Guruswami and Sudan in the case of Reed-Solomon codes. In particular, we notice that matrices that occur in the context of list decoding have low displacement rank, and use this fact to derive algorithms that use  $O(n^2\ell)$  and  $O(n^{7/3}\ell)$  operations over the base field for list decoding of Reed-Solomon codes and algebraic-geometric codes from certain plane curves, respectively. Here  $\ell$  denotes the length of the list; assuming that  $\ell$  is constant, this gives algorithms of running time  $O(n^2)$  and  $O(n^{7/3})$ , which is the same as the running time of conventional decoding algorithms. We also present efficient parallel algorithms for the above tasks.

To the best of our knowledge this is the first application of the concept of displacement rank to the unified derivation of several decoding algorithms; the technique can be useful in finding efficient and fast methods for solving other decoding problems.

## 1. Introduction

Matrices with different patterns of structure are frequently encountered in the context of coding theory. For example, Hankel matrices  $H = [h_{i-j}]$  arise in the Berlekamp-Massey algorithm [3], Vandermonde matrices  $V = [x_i^j]$  are encountered

---

1991 *Mathematics Subject Classification*. Primary: 11T71, 94B99, Secondary: 15A57.

*Key words and phrases.* Reed-Solomon codes, algebraic codes, displacement structure.

This work was supported in part by NSF contracts CCR 0098222 and 0242518.

in the context of Reed-Solomon codes, and Cauchy matrices  $C = [\frac{1}{x_i - y_j}]$  are related to the classical Goppa codes [32]. There are quite a few other examples involving certain generalizations of such matrices, e.g., paired Vandermonde matrices  $V = [V|DV]$  with diagonal  $D$  and Vandermonde  $V$  occur in the context of the Welch-Berlekamp algorithm [4]. In most cases one is interested in finding a nonzero element in the kernel of the matrix. This problem has been solved for each of the above cases resulting in existing efficient algorithms. Although it is obvious that these algorithms make (often implicit) use of the structure of the underlying matrices, in each case this exploitation was limited to a particular pattern of structure.

Structured matrices appear in many other areas as well, and in some of these areas there are unified methods to derive efficient formulas and algorithms for them. Since we focus here on a particular application, it is virtually impossible to give an adequate introduction to all existing approaches here. We refer an interested reader to [36] for a survey of a rational matrix interpolation approach, to [8] for the description of the use of the concept of the reproducing kernels. Their techniques can also be used to derive efficient decoding algorithms. In this paper we limit ourselves to another well-known approach based on the concept of displacement structure (we provide some historical notes below). The latter method may have certain advantages over the others since it can be described using transparent linear algebra terms as follows. Let two auxiliary simple (e.g., Jordan form) matrices  $F$  and  $A$  be given and fixed, and let  $R$  belongs to the class of matrices with the low displacement rank, the latter is defined as

$$\alpha = \text{rank}(FR - RA).$$

Then one can factor (non-uniquely)

$$(1) \quad FR - RA = GB^T, \quad (G, B \in \mathbb{C}^{n \times \alpha}).$$

i.e., both matrices on the right-hand side of (1) have only  $\alpha$  columns each. Thus, the displacement rank  $\alpha$  measures the complexity of  $R$ , because if (1) is solvable for  $R$ , then all its  $n^2$  entries are described by only  $2\alpha n$  entries of  $\{G, B\}$  (in most of applications the auxiliary matrices  $F$  and  $A$  are fixed and simple). The displacement structure approach is in solving matrix problems for  $R$  by ignoring its entries and using instead only  $2\alpha n$  entries of  $\{G, B\}$ . Operating on  $\alpha n$  parameters rather than on  $n^2$  entries allows one to achieve computational savings. There are various types of efficient algorithms (such as inversion, Levinson-type, Schur-type, mixed, etc.) that can be derived via this approach. Until now we did not address specific patterns of structure. Several easily verified inequalities shown in Table 1 explain why the approach has been found to be useful to study Hankel, Vandermonde or Cauchy matrices. For example, for Cauchy matrices  $R = [\frac{1}{x_i - y_j}]$  we have:  $\text{rank}(D_x R - RD_y) = \text{rank}[\frac{x_i - y_j}{x_i - y_j}] = \text{rank}[1] = 1$ . We see that each pattern of structure in Table 1 is associated with its own choice of the auxiliary matrices  $\{F, A\}$  in (1); the name “displacement structure” was introduced since the approach was originally used only for Toeplitz matrices for which the auxiliary matrices are *displacement (or shift)* matrices  $Z$ . In particular, in the very first publications on this subject it was noticed that in many instances there is no difference between the classical Toeplitz matrices for which the displacement rank is 2 and the more general Toeplitz-like matrices. The latter are defined as those with a small (though possibly bigger than

Toeplitz $R = [t_{i-j}]$	$\alpha = \text{rank } (ZR - RZ)$	$\leq 2$
Hankel $R = [h_{i+j}]$	$\alpha = \text{rank } (ZR - RZ^T)$	$\leq 2$
Vandermonde $R = [x_i^j]$	$\alpha = \text{rank } (D_x^{-1}R - RZ^T)$	$= 1$
Cauchy $R = [\frac{1}{x_i - y_j}]$	$\alpha = \text{rank } (D_x R - RD_y)$	$= 1$

TABLE 1. Here  $Z^T = J(0)$  is one Jordan block with the eigenvalue 0, and  $\{D_x, D_y\}$  are diagonal matrices.

2) displacement rank with respect to the *shift* (or *displacement*) auxiliary matrices  $F = A = Z$ . We next give some references.

Perhaps the first reference explicitly discussing efficient algorithms for matrices with displacement structure is the thesis [33] where in Sec. 4.0.1 M.Morf writes: “*In October 1970 the crucial shift-low-rank updating property was recognized by the author as the proper generalization of the Toeplitz and Hankel structure of matrices... The algorithm was called “Fast Cholesky decomposition” (or RMH4, the forth in a series of related algorithms, see Sec 6.1). In June 1971 computer programs for both scalar and block decompositions were successfully completed by the author. It was found that the known numerical well behavior of the Cholesky decomposition seemed to be inherited.*” Thesis [33] contains a plethora of ideas and interesting references; however it is not widely accessible, and unfortunately it is sometimes improperly referenced in recent reviews. The birth certificate to the concept of displacement structure was given in journal publications [9] and [23]. However, let us also mention the paper [42] (see also [43]), where displacement equations appeared in the context of factorization of rational matrix functions, and where the crucial property of the Schur complements of matrices with displacement structure was first noticed; matrix interpretations of some Sakhnovich’s results were independently derived in [34]. For completeness let us also mention a few papers dealing with continuous analogs of the concept of displacement. In [40, 41] the idea of displacement structure in principle appears for integral operators with a difference kernel. In [21] this idea appeared in connection with the Chandrasekhar equations. We finally note that the displacement structure considered in [33] and [9] was limited to Toeplitz-like matrices. A general displacement approach was proposed in [16] and was applied in [17] to study matrices related not only to Toeplitz but also to Vandermonde and Cauchy matrices. For more information and plethora of different results and approaches we refer also to [17], [27], [8], [36] and [35] and (complementing each other) references therein.

Though the displacement approach to developing fast algorithms for structured matrices is well-understood, it has never been applied to unify the derivation of efficient decoding algorithms. In this paper we show how to use it to derive in a unified manner a number of efficient decoding algorithms. We first derive solutions to list-decoding problems concerning Reed-Solomon- and algebraic-geometric codes. Given a received word and an integer  $e$ , a list decoding algorithm returns a list of all codewords which have distance at most  $e$  from the received word. Building on a sequence of previous results [46, 5, 1], Sudan [45] was the first to invent an efficient list-decoding algorithm for Reed-Solomon-codes. This algorithm, its subsequent generalizations by Shokrollahi and Wasserman [44] to algebraic-geometric codes, and the recent extension by Guruswami and Sudan [15] are among the best decoding

algorithms known in terms of the number of errors they can correct. All these algorithms run in two steps. The first step, which we call the *linear algebra* step, consists of computing a nonzero element in the kernel of a certain structured matrix. This element is then interpreted as a polynomial over an appropriate field: the second step, called the *root-finding step* tries to find the roots of this polynomial over that field. The latter is a subject of investigation of its own and can be solved very efficiently in many cases [2, 11, 39], so we will concentrate in this paper on the first step only. This will be done by applying our general algorithm in Sections 4 and 5. Specifically, we will for instance easily prove that the linear algebra step of decoding Reed-Solomon-codes of block length  $n$  with lists of length  $\ell$  can be accomplished in time  $O(n^2\ell)$ . A similar time bound holds also for the root-finding step, and so the overall running time of the list-decoding procedure is of this order of magnitude. This result matches that of Roth and Ruckenstein [39], though the latter has been obtained using completely different methods. Furthermore, we will design a novel  $O(n^{7/3}\ell)$  algorithm for the linear algebra step of list decoding of certain algebraic-geometric-codes from plane curves of block length  $n$  with lists of length  $\ell$ . We remark that, using other means, Høholdt and Refslund Nielsen [18] have obtained an algorithm for list decoding on Hermitian curves which solves both steps of the algorithm in [15] more efficiently.

In comparison to the existing decoding algorithms, the displacement structure approach seems to have the advantage of leading to a transparent algorithm design. For instance, it allowed us to design parallel decoding algorithms by using the same principles as those of sequential algorithms, though the details are more complicated since one needs have to modify the corresponding types of structure.

The paper is organized as follows. Sections 2 and 3 recall basic definitions of the displacement structure theory, and specify several particular versions of matrix algorithms to be used and analyzed in the rest of the paper. Sections 4, 5, and 6 apply these results to speed up Sudan's algorithm, its generalization by Shokrollahi-Wasserman, and the improvement of Guruswami-Sudan, respectively. The running times for these algorithms range from  $O(\ell n^2)$  for Reed-Solomon codes to  $O(n^{7/3}\ell)$  for AG-codes on curves from plane algebraic curves, where  $n$  is the block-length of the code and  $\ell$  is the list-size. Section 7 introduces methods for the construction of efficient parallel algorithms for the above tasks.

## 2. The Displacement Structure Approach

The problem of computing a nonzero element  $x$  in the kernel  $Vx = 0$  of a certain structured matrix  $V$  frequently occurs in the context of constructing decoding algorithms. This is a trivial linear algebra problem; however exploiting the structure of the underlying matrix typically allows one to achieve computational savings. In this paper we solve such problem for the three kinds structured matrices (10), (13), (14) shown in the main text below. Before addressing particular details for each of these matrices in sec. 4, 5 and 6, resp. we specify in this section some general frameworks for finding kernel vectors for these matrices, the latter task can be solved by using Gaussian elimination for computing a *PLU*-decomposition  $V = PLU$  with a permutation matrix  $P$ , an invertible lower triangular matrix  $L$ , and an upper triangular matrix  $U$ . We would like to give a succinct presentation of this well-known procedure, as it is necessary to set up the notations and it is useful for the understanding of the algorithms presented below.

It is well-known (cf. with [10]) that applying Gaussian elimination to an arbitrary matrix  $V$  is equivalent to recursive Schur complementation, as shown by factorization

$$(2) \quad V = \begin{pmatrix} v & V_{12} \\ V_{21} & V_{22} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ V_{21}\frac{1}{v} & I \end{pmatrix} \cdot \begin{pmatrix} v & V_{12} \\ 0 & V_2 \end{pmatrix},$$

where  $V_2 = V_{22} - V_{21}\frac{1}{v}V_{12}$  is the Schur complement of the (1,1) entry  $v$  in the matrix  $V$ . Applying (2) recursively to compute  $V_2 = L_2U_2$  one obtains the  $LU$  factorization for the entire matrix  $V$ :

$$V = \underbrace{\begin{pmatrix} 1 & 0 \\ V_{21}/v & L_2 \end{pmatrix}}_{=:L} \cdot \underbrace{\begin{pmatrix} v & V_{21} \\ 0 & U_2 \end{pmatrix}}_{=:U}.$$

This is the basis for the Gaussian elimination algorithm, which iteratively computes Schur-complements and an  $LU$ -decomposition thereof. Throughout the above discussion we assumed  $v \neq 0$  which is not always true. This can be easily resolved by pivoting (row interchanges), so that the algorithm additionally produces a permutation matrix  $Q$  such that  $QV$  has the property that the (1,1)-entry of all the Schur-complements are nonzero, and hence the above procedure is applicable. Altogether, the algorithm produces a decomposition  $V = PLU$ , where  $P = Q^{-1}$ .

Note that it is trivial to compute a nonzero element in the kernel of  $V$  once a  $PLU$ -decomposition is known, since the kernel of  $V$  and that of  $U$  coincide, and a nonzero element in the latter can be found using backward substitution.

It is now well known that the above procedure can be speeded up in the case matrix  $V$  has a displacement structure as in

$$\begin{pmatrix} d_1 & \mathbf{0} \\ * & D_2 \end{pmatrix} V - V \begin{pmatrix} a_1 & * \\ \mathbf{0} & A_2 \end{pmatrix} = G_1 B_1$$

where the so-called *generator matrices*  $G_1 \in K^{m \times \alpha}$  and  $B_1 \in K^{\alpha \times n}$  are rectangular, so that the displacement rank of  $V$  is  $\leq \alpha$ . As was noticed by Sakhnovich [42] (see also [43] and the relevant description of his results in [14] and [13]) in this case the Schur complement  $V_2 = V_{22} - V_{21}\frac{1}{v}V_{12}$  also has the displacement rank  $\leq \alpha$ , so that

$$D_2 V_2 - V_2 A_2 = G_2 B_2,$$

for some new generator matrices  $G_2 \in K^{m \times \alpha}$  and  $B_2 \in K^{\alpha \times n}$  (also containing only  $2\alpha n$  entries). Sakhnovich was not interested in computational issues, and he used the above result for factorizing rational matrix functions. M.Morf [34] (and simultaneously [6]) independently arrived at a similar observation but they used them to derive a fast algorithm for factoring Toeplitz matrices. The technique (we suggest to call it Sakhnovich-Morf principle) is based on replacing update of  $n^2$  elements in  $V \rightarrow V_2$  by updating of only  $2\alpha n$  elements in  $\{G_1, B_1\} \rightarrow \{G_2, B_2\}$ . In fact they applied a divide-and-conquer technique to obtain a superfast algorithm. Their target was Toeplitz matrices but the proofs go through without much modifications for the other patterns of structure as well.

Since then a number of variants of formulas were used for computing  $\{G_1, B_1\} \rightarrow \{G_2, B_2\}$ . A recent survey paper [27] can be consulted, however it covers only a subset of the relevant literature, and the reader may wish to complement it with reading recent surveys [36] and [8] covering more relevant literature. It is also worth to look more closely at the early references [33], [7], [30] (see also a relevant discussion in [31]).

In this paper we use one of the variants of the updating formulas  $\{G_1, B_1\} \longrightarrow \{G_2, B_2\}$  obtained in [13] and [14] and recalled next.

LEMMA 2.1. *Let*

$$(3) \quad \begin{pmatrix} D_1 & \mathbf{0} \\ * & D_2 \end{pmatrix} \begin{pmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{pmatrix} - \begin{pmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{pmatrix} \begin{pmatrix} A_1 & * \\ \mathbf{0} & A_2 \end{pmatrix} = \begin{pmatrix} g_{11} \\ G_{21} \end{pmatrix} (b_{11} \mid B_{12})$$

*Suppose that  $v_{11}$  is nonzero. Then the Schur complement  $V_2 := V_{22} - V_{21}V_{11}^{-1}V_{12}$  of  $V$  satisfies the equation*

$$D_2 V_2 - V_2 A_2 = G_2 B_2,$$

where

$$(4) \quad G_2 = G_{21} - V_{12}V_{11}^{-1}g_{11}, \quad B_2 = B_{12} - b_{11}V_{11}^{-1}V_{12}.$$

The above lemma is a matrix part of a more general result on factorization of rational matrix functions [13], [14]. If one focuses on this matrix part only, it admits a trivial proof (cf. with [24]).

PROOF. Indeed, from (3) and the standard Schur complementation formula

$$V = \begin{pmatrix} I & \mathbf{0} \\ V_{21}V_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} V_{11} & \mathbf{0} \\ \mathbf{0} & V_2 \end{pmatrix} \begin{pmatrix} I & V_{11}^{-1}V_{12} \\ \mathbf{0} & I \end{pmatrix}.$$

it follows that

$$\begin{aligned} & \begin{pmatrix} D_1 & \mathbf{0} \\ * & D_2 \end{pmatrix} \begin{pmatrix} V_{11} & \mathbf{0} \\ \mathbf{0} & V_2 \end{pmatrix} - \begin{pmatrix} V_{11} & \mathbf{0} \\ \mathbf{0} & V_2 \end{pmatrix} \begin{pmatrix} A_1 & * \\ \mathbf{0} & A_2 \end{pmatrix} = \\ & \begin{pmatrix} 1 & \mathbf{0} \\ -V_{21}V_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} g_{11} \\ G_{21} \end{pmatrix} (b_{11} \mid B_{12}) \begin{pmatrix} 1 & -V_{11}^{-1}V_{12} \\ \mathbf{0} & I \end{pmatrix}. \end{aligned}$$

Equating the (2.2) block entries, one obtains (4).  $\square$

The above simple lemma will be used to derive algorithms for processing several structured matrices appearing in the context of three decoding problems. Before addressing a number of issues on implementing the above result to solve these problems we make the following comment.

REMARK 2.2. *It is obvious that the latter lemma can be used to replace the update of  $n^2$  elements in  $V \longrightarrow V_2$  by updating of only  $2\alpha n$  elements in  $\{G_1, B_1\} \longrightarrow \{G_2, B_2\}$ , and to continue recursively. One can ask how should we compute the very first generator  $\{G_1, B_1\}$  which is the input of the suggested algorithm. In fact, in almost all applications we know about the matrix itself is never given: what is given is a generator of this matrix. Hence no preprocessing computations are necessary. Many examples can be given, we could suggest the reader to jump to the middle of our paper and to look at the matrix in (14) appearing in the context of the Guruswami-Sudan algorithm. For this matrix its generator  $\{H, U\}$  on the right hand side of (16) is immediately given by (18) and (19). So the recursive generator recursion  $\{H, U\} \longrightarrow \{H_2, H_2\}$  is possible without any preprocessing computations.*

Before providing the exact record of the algorithm based on lemma 2.1 we need to clarify one issue. If  $V$  has a nonzero entry in its first column, say at position  $(k, 1)$ , then we can consider  $PV$  instead of  $P$ , where  $P$  is the matrix corresponding to interchanging rows 1 and  $k$ . The displacement equation for  $PV$  is then given by  $(PDP^\top)(PV) - PVA = PGB$ . In order to apply the previous lemma, we need to

ensure that  $PDP^T$  is lower triangular. But since  $P$  can be any transposition, this implies that  $D$  is a diagonal matrix.

It is possible to use the above lemma to design an algorithm for computing a  $PLU$ -decomposition of the matrix  $V$ , see [37]. Note that it is trivial to compute a nonzero element in the kernel of  $V$  once a  $PLU$ -decomposition is known, since the kernel of  $V$  and that of  $U$  coincide, and a nonzero element in the latter can be found using backward substitution.

**ALGORITHM 2.3.** *On input a diagonal matrix  $D \in K^{m \times m}$ , an upper triangular matrix  $A \in K^{n \times n}$ , and a  $\nabla_{D,A}$ -generator  $(G, B)$  for  $V \in K^{m \times n}$  in  $DV - VA = GB$ , the algorithm outputs a permutation matrix  $P$ , a lower triangular matrix  $L \in K^{m \times m}$ , and an upper triangular matrix  $U \in K^{m \times n}$ , such that  $V = PLU$ .*

- (1) Recover from the generator the first column of  $V$ .
- (2) Determine the position, say  $(k, 1)$ , of a nonzero entry of  $V$ . If it does not exist, then set the first column of  $L$  equal to  $[1, \mathbf{0}]^\top$  and the first row of  $U$  equal to the first row of  $V$ , and go to Step (4). Otherwise interchange the first and the  $k$ -th diagonal entries of  $A$  and the first and the  $k$ -th rows of  $G$  and call  $P_1$  the permutation matrix corresponding to this transposition.
- (3) Recover from the generator the first row of  $P_1 V =: \begin{pmatrix} v_{11} & V_{12} \\ V_{21} & V_{22} \end{pmatrix}$ . Store  $[1, V_{12}/v_{11}]^\top$  as the first column of  $L$  and  $[v_{11}, V_{12}]$  as the first row of  $U$ .
- (4) If  $v_{11} \neq 0$ , compute by Lemma 2.1 a generator of the Schur complement  $V_2$  of  $P_1 V$ . If  $v_{11} = 0$ , then set  $V_2 := V_{22}$ ,  $G_2 := G_{21}$ , and  $B_2 := B_{12}$ .
- (5) Proceed recursively with  $V_2$  which is now represented by its generator  $(G_2, B_2)$  to finally obtain the factorization  $V = PLU$ , where  $P = P_1 \cdots P_\mu$  with  $P_k$  being the permutation used at the  $k$ -th step of the recursion and  $\mu = \min\{m, n\}$ .

The correctness of the above algorithm and its running time depend on steps (1) and (3). Note that it may not be possible to recover the first row and column of  $V$  from the matrices  $D, A, G, B$ . We will explore these issues later in Section 3.

For now, we focus on developing a more memory efficient version of this algorithm for certain displacement structures. For this, the following result will be crucial.

**LEMMA 2.4.** *Let  $V \in K^{m \times n}$  be partitioned as*

$$V = \begin{pmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{pmatrix},$$

*for some  $1 \leq i < m$ , where  $V_{11} \in K^{i \times i}$ ,  $V_{12} \in K^{i \times (n-i)}$ ,  $V_{21} \in K^{(m-i) \times i}$ , and  $V_{22} \in K^{(m-i) \times (m-i)}$ , and suppose that  $V_{11}$  is invertible. Further, let  $\tilde{V} := \begin{pmatrix} V \\ I_n \end{pmatrix}$ , where  $I_n$  is the  $n \times n$ -identity matrix, and denote by  $x = (x_1, \dots, x_{m+n-i})^\top$  the first column of the Schur complement of  $\tilde{V}$  with respect to  $V_{11}$ . Suppose that  $x_1 = \dots = x_{m-i} = 0$ . Then the vector  $(x_{m-i+1}, \dots, x_m, 1, 0, \dots, 0)^\top$  is in the right kernel of the matrix  $V$ .*

**PROOF.** The Schur complement of  $\tilde{V}$  with respect to  $V_{11}$  is easily seen to be

$$\begin{pmatrix} V_{22} - V_{21}V_{11}^{-1}V_{12} \\ -V_{11}^{-1}V_{12} \\ I_{n-i} \end{pmatrix}.$$

Note that

$$V \cdot \begin{pmatrix} -V_{11}^{-1}V_{12} \\ I_{n-i} \end{pmatrix} = \begin{pmatrix} 0 \\ V_{22} - V_{11}^{-1}V_{12} \end{pmatrix}.$$

By assumption, the first column of the matrix on the right is zero, which implies the assertion.  $\square$

Suppose now that  $V$  has low displacement rank with respect to  $\nabla_{D,Z}$  and suppose further that  $A$  is a matrix such that  $A - Z$  has low rank:

$$DV - VZ = G_1B_1, \quad A - Z = G_2B_2.$$

Then we have (cf. with [22])

$$(5) \quad \begin{pmatrix} V & 0 \\ 0 & A \end{pmatrix} \cdot \begin{pmatrix} V \\ I_n \end{pmatrix} - \begin{pmatrix} V \\ I_n \end{pmatrix} Z = \begin{pmatrix} G_1 & 0 \\ 0 & G_2 \end{pmatrix} \cdot \begin{pmatrix} B_1 \\ B_2 \end{pmatrix}.$$

Algorithm 2.3 can now be customized in the following way.

**ALGORITHM 2.5.** *On input a diagonal matrix  $D \in K^{m \times m}$ , an upper triangular matrix  $Z \in K^{n \times n}$ , a matrix  $A \in K^{n \times n}$ , and a  $\nabla_{C,Z}$ -generator  $(G, B)$  for  $W = \begin{pmatrix} V \\ I_n \end{pmatrix} \in K^{(m+n) \times n}$  in  $CW - WZ = GB$ , where  $C = \begin{pmatrix} D & 0 \\ 0 & A \end{pmatrix}$ , the algorithm outputs a vector  $x = (x_1, \dots, x_i, 1, 0, \dots, 0)^\top$  such that  $Vx = 0$ .*

- (0) Set  $i := 0$ .
- (1) Recover from the generators the first column of  $W$ .
- (2) Determine the position, say  $(k, 1)$ , of a nonzero entry of  $W$ . If  $k$  does not exist, or  $k > m - i$ , then go to Step (6). Otherwise interchange the first and the  $k$ -th diagonal entries of  $D$ , the first and the  $k$ -th rows of  $G$ , the first and the  $k$ -th entries of the first column of  $W$ , denoting the new matrix by  $W$  again.
- (3) Recover from the generators the first row of  $W$ .
- (4) Using the first row and the first column of  $W$ , compute by Lemma 2.1 a generator of the Schur complement  $W_2$  of  $W$ .
- (5) Proceed recursively with  $W_2$  which is now represented by its generator  $(G_2, B_2)$ , increase  $i$  by 1, and go back to Step (1).
- (6) Output the vector  $(x_1, \dots, x_i, 1, 0, \dots, 0)^\top$  where  $x_1, \dots, x_i$  are the  $m - i + 1$ -st through  $m$ -th entries of the first column of  $W$ .

As was pointed out before, the correctness of the above algorithm and its running time depend on steps (1) and (3). Note that it may not be possible to recover the first row and column of  $W$  from the matrices  $D, A, G, B, Z$ . In fact, recovery from these data alone is only possible if  $\nabla = \nabla_{C,Z}$  is an isomorphism. For simplicity we assume in the following that this is the case. In the general case one has to augment the  $(D, A, G, B, Z)$  by more data corresponding to the kernel of  $\nabla$ , see [36, Sect. 5] and Appendix A.

If  $\nabla$  is an isomorphism, then the algorithm has the correct output. Indeed, using Lemma 2.1, we see that at step (5),  $W_2$  is the Schur-complement of the matrix  $PW$  with respect to the principal  $i \times i$ -minor, where  $P$  is a permutation matrix (obtained from the pivoting transpositions in Step (2)). Once the algorithm reaches Step (6), the conditions of Lemma 2.4 are satisfied and the algorithm outputs the correct vector.

The running time of the algorithm is summarized in the following.

LEMMA 2.6. Suppose that steps (1) and (3) of Algorithm 2.5 run in time  $O(\alpha m)$  and  $O(\alpha n)$ , respectively. Then the total running time of that algorithm is  $O(\alpha mn)$ , where  $\alpha$  is the displacement rank of  $V$  with respect to  $\nabla_{D,A}$ .

PROOF. The proof is obvious once one realizes that Step (4) runs in time  $O(\alpha(m+n))$ , and that the algorithm is performed recursively at most  $\min\{m, n\}$  times.  $\square$

### 3. Computing the First Row and the First Column

A major ingredient of Algorithm 2.5 is that of computing the first row and the first column of a matrix from its generators. The computational cost of this step depends greatly on the underlying displacement structure. In this section, we will present a solution to this problem in a special case, namely, in the case where the matrix  $W \in K^{(m+n) \times n}$  has the displacement structure

$$(6) \quad \begin{pmatrix} D & 0 \\ 0 & A \end{pmatrix} W - WZ = GB$$

where  $D$  is diagonal and  $Z, A \in K^{n \times n}$  are given by

$$(7) \quad Z := \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & 0 & \cdots & 0 & 0 \end{pmatrix}, \quad A := \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & 0 \end{pmatrix}.$$

$Z$  is called the *upper shift matrix* of format  $n$ . This case will be prototypical for our later applications. For the sake of simplicity, we will assume that all the diagonal entries of  $D$  are nonzero.

ALGORITHM 3.1. On input a diagonal matrix  $D$  with nonzero diagonal entries  $x_1, \dots, x_m$ , a matrix  $G = (G_{ij}) \in K^{m \times \alpha}$ , and a matrix  $B = (B_{ij}) \in K^{\alpha \times n}$ , this algorithm computes the first row and the first column of the matrix  $W \in K^{(m+n) \times n}$  which satisfies the displacement equation (6).

- (1) Compute the first row  $(r_1, \dots, r_n)$  and the first column  $(\gamma_1, \dots, \gamma_m, \gamma_{m+1}, \dots, \gamma_{m+n})^\top$  of  $GB$ .
- (2) For  $i = 1, \dots, m$  set  $v_i := \gamma_i/x_i$ .
- (3) For  $i = m+1, \dots, m+n-1$  set  $v_{i+1} := \gamma_i$ .
- (4) Set  $v_{m+1} := \gamma_{m+n}$ .
- (5) Set  $c_1 := v_1$ . For  $i = 2, \dots, n$  set  $c_i := (r_i + c_{i-1})/x_i$ .
- (6) Output row  $(c_1, \dots, c_n)$  and column  $(v_1, \dots, v_m, v_{m+1}, \dots, v_{m+n})^\top$ .

PROPOSITION 3.2. The above algorithm correctly computes its output with  $O(\alpha(m+n))$  operations in the field  $K$ .

PROOF. Correctness of the algorithm follows immediately from the following observation: let  $(c_1, \dots, c_n)$  and  $(c_1, v_2, \dots, v_m, v_{m+1}, \dots, v_{m+n})^\top$  be the first row

and the first column of  $W$ , respectively. Then

$$\begin{pmatrix} D & 0 \\ 0 & A \end{pmatrix} W - WZ = \begin{pmatrix} x_1 c_1 & x_1 c_2 - c_1 & \cdots & x_1 c_n - c_{n-1} \\ x_2 v_2 & * & \cdots & * \\ \vdots & \vdots & \ddots & \vdots \\ x_m v_m & * & \cdots & * \\ v_{m+2} & * & \cdots & * \\ \vdots & \vdots & \cdots & \vdots \\ v_{m+n} & * & \ddots & * \\ v_{m+1} & * & \cdots & * \end{pmatrix}.$$

where the  $*$ 's denote elements in  $K$ . As for the running time, observe first that computing the first row and first column of  $GB$  requires at most  $\alpha(m+n) + \alpha n$  operations over  $K$ . Step (2) can be done using  $m$  operations, steps (3) and (4) are free of charge, and Step (5) requires  $2n$  operations.  $\square$

We can now combine all the results obtained so far to develop an efficient algorithm for computing a nonzero element in the kernel of a matrix  $V \in K^{m \times n}$  given indirectly as a solution to the displacement equation

$$(8) \quad DV - VZ = GB.$$

Here,  $D$  is a diagonal matrix,  $Z$  is an upper-shift matrix of format  $n$ ,  $G \in K^{m \times \alpha}$ , and  $B \in K^{\alpha \times n}$ . Since we will use Algorithm 2.5, we need to have a displacement structure for  $W = \begin{pmatrix} V \\ I_n \end{pmatrix}$ . It is given by

$$(9) \quad \begin{pmatrix} D & 0 \\ 0 & A \end{pmatrix} \begin{pmatrix} V \\ I_n \end{pmatrix} - \begin{pmatrix} V \\ I_n \end{pmatrix} Z = \begin{pmatrix} G \\ C \end{pmatrix} B,$$

where  $A$  is defined as in (7) and  $C = A - Z$ . We assume that none of the diagonal entries of  $D$  are zero.

**ALGORITHM 3.3.** *On input integers  $m, n$ ,  $m < n$ , a diagonal matrix  $D \in K^{m \times m}$  all of whose diagonal entries  $x_1, \dots, x_m$  are nonzero, a matrix  $G \in K^{m \times \alpha}$ , and a matrix  $B \in K^{\alpha \times n}$ , the algorithm computes a nonzero vector  $v \in K^n$  in the kernel of the matrix  $V$  given by the displacement equation (8).*

- (1) Set  $G_1 := \begin{pmatrix} G \\ A-Z \end{pmatrix}$ ,  $B_1 := B$ , and  $D_1 := D$ , where  $A$  is given in (7).
- (2) For  $i = 1, \dots, m$  do
  - (a) Let  $D_i$  be the diagonal matrix with diagonal entries  $x_i, \dots, x_m$ . Use Algorithm 3.1 with input  $D_i$ ,  $G_i$ , and  $B_i$  to compute the column  $c = (c_1, \dots, c_{m+n-i+1})^\top$ .
  - (b) If  $c_1 = \dots = c_{m-i+1} = 0$ , then **output**  $(c_{m-i+2}, \dots, c_n, 1, 0, \dots, 0)^\top$  and **stop**. Otherwise, find an integer  $k \leq m-i+1$  such that  $c_k \neq 0$ . Interchange rows  $k$  and  $1$  of  $c$  and of  $G_i$ , interchange the first and the  $k$ th diagonal entries of  $D_i$ .
  - (c) Use Algorithm 3.1 with input  $D_i$ ,  $G_i$  and  $B_i$  to compute the row  $r = (r_1, \dots, r_{n-i+1})$ .
  - (d) For  $j = 2, \dots, m+n-i+1$  replace row  $j$  of  $G_i$  by  $-c_j/c_1$  times the first row plus the  $j$ -th row. Set  $G_{i+1}$  as the matrix formed from  $G_i$  by deleting the first row.

- (e) For  $j = 2, \dots, n-i+1$  replace the  $j$ -th column of  $B_i$  by  $-r_j/c_1$  times the first column plus the  $j$ -th column. Set  $B_{i+1}$  as the matrix formed from  $B_i$  by deleting the first column.

**THEOREM 3.4.** *The above algorithm correctly computes its output with  $O(\alpha mn)$  operations over the field  $K$ .*

**PROOF.** The correctness of the algorithm follows from Lemma 2.4 and Proposition 3.2. The analysis of the running time of the algorithm is straight-forward: Step (2a) runs in time  $O(\alpha(m+n-i))$  by Proposition 3.2. The same is true for steps (2d) and (2e). Hence, in the worst case, the running time will be

$$\sum_{i=1}^{m-1} O(\alpha(m+n-i+1)) = O(\alpha mn), \text{ since } m < n. \quad \square$$

The memory requirements for this algorithm can be reduced by observing that the last  $n-i$  rows of the matrix  $G_i$  are always known. As a result, the matrix  $G_i$  needs to store only  $m(\ell+1)$  elements (instead of  $(m+n)(\ell+1)$ ). Further, it is easy to see that for the vector  $c$  computed in Step (2a), we have  $c_{m+1} = 1, c_{m+2} = \dots = c_{m+n-i} = 0$ , hence, we do not need to store these results. Combining these observations, one can easily design an algorithm that needs storage for two matrices of sizes  $m \times (\ell+1)$  and  $(\ell+1) \times n$ , respectively, and storage for three vectors of size  $n$ .

#### 4. The Algorithm of Sudan

In [45] Sudan describes an algorithm for list decoding of RS-codes which we will briefly describe here. Let  $\mathbb{F}_q[x]_{<k}$  denote the space of polynomials over the finite field  $\mathbb{F}_q$  of degree less than  $k$ , and let  $x_1, \dots, x_n$  denote distinct elements of  $\mathbb{F}_q$ , where  $k \leq n$ . The image of the morphism  $\gamma: \mathbb{F}_q[x]_{<k} \rightarrow \mathbb{F}_q^n$  mapping a polynomial  $f$  to the vector  $v := (f(x_1), \dots, f(x_n))$  is a linear code over  $\mathbb{F}_q$  of dimension  $k$  and minimum distance  $n-k+1$ . Suppose the vector  $v$  is sent over a communication channel and the vector  $u := (y_1, \dots, y_n)$  is received. If the Hamming distance between  $u$  and  $v$  is at most  $(n-k)/2$ , then conventional decoding algorithms like the Berlekamp-Massey algorithm can decode  $u$  to the correct codeword  $v$ . If the number  $e$  of errors is larger than  $(n-k)/2$ , then Sudan's algorithm compiles a list of at most  $\ell = \ell(e)$  codewords which contains  $v$ . The algorithm consists of two steps. Let  $b := \lfloor n/(\ell+1) + \ell(k-1)/2 + 1 \rfloor$ . The first step uses Lagrange interpolation to compute a bivariate polynomial  $F = \sum_{i=0}^{\ell} F_i(x)y^i$  with  $\deg F_i < b - i(k-1)$  and such that  $F(x_t, y_t) = 0$  for all  $t = 1, \dots, n$ . The second step computes the roots of  $F$  which are of the form  $y - g(x)$ ,  $g \in \mathbb{F}_q[x]_{<k}$ , and outputs those  $g$  such that  $\gamma(g)$  and  $u$  have distance at most  $e$ . The relationship between  $\ell$  and  $e$  is given indirectly by  $e \leq n-b$ . There are efficient algorithms for solving the second step [2, 11, 39]. In the following we will concentrate on the problem of efficiently computing the polynomial  $F$ .

This polynomial corresponds to a nonzero element in the kernel of a matrix  $V$  with a repetitive structure which we will describe below. Let  $d_0, \dots, d_\ell \geq 0$  be

defined by  $d_i := b - i(k - 1)$  for  $0 \leq i < \ell$ , and  $d_\ell = n + 1 - \sum_{i=0}^{\ell-1} d_i$ . Let  $V := (10)$

$$\begin{pmatrix} 1 & x_1 & \cdots & x_1^{d_0-1} & y_1 & y_1x_1 & \cdots & y_1x_1^{d_1-1} & \cdots & y_1^\ell & y_1^\ell x_1 & \cdots & y_1^\ell x_1^{d_\ell-1} \\ 1 & x_2 & \cdots & x_2^{d_0-1} & y_2 & y_2x_2 & \cdots & y_2x_2^{d_1-1} & \cdots & y_2^\ell & y_2^\ell x_2 & \cdots & y_2^\ell x_2^{d_\ell-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \cdots & x_n^{d_0-1} & y_n & y_nx_n & \cdots & y_nx_n^{d_1-1} & \cdots & y_n^\ell & y_n^\ell x_n & \cdots & y_n^\ell x_n^{d_\ell-1} \end{pmatrix},$$

and let  $v = (v_{00}, v_{01}, \dots, v_{0,d_0-1}, \dots, v_{\ell,0}, v_{\ell,1}, \dots, v_{\ell,d_\ell-1}) \in \mathbb{F}_q^{n+1}$  be a nonzero vector such that  $V \cdot v = 0$ . Then the polynomial  $F(x, y) = \sum_{i=0}^\ell F_i(x)y^i$  where  $F_i(x) = v_{i,0} + v_{i,1}x + \cdots + v_{i,d_i-1}x^{d_i-1}$  satisfies  $F(x_t, y_t) = 0$  for  $t = 1, \dots, n$ . The task at hand is thus to compute a nonzero element in the kernel of  $V$ . Using the displacement approach we can easily compute such a vector in time  $O(n^2\ell)$ . For constant list-size  $\ell$  we thus obtain an algorithm whose running time is quadratic in  $n$ .

Let us first prove that  $V$  has displacement rank at most  $\ell+1$  with respect to suitable matrices. Let  $D$  be the diagonal matrix with diagonal entries  $1/x_1, \dots, 1/x_n$ . Define

$$(11) \quad G := \begin{pmatrix} 1/x_1 & y_1/x_1 - x_1^{d_0-1} & \cdots & y_1^{\ell-1}(y_1/x_1 - x_1^{d_{\ell-1}-1}) \\ 1/x_2 & y_2/x_2 - x_2^{d_0-1} & \cdots & y_2^{\ell-2}(y_2/x_2 - x_2^{d_{\ell-1}-1}) \\ \vdots & \vdots & \ddots & \vdots \\ 1/x_n & y_n/x_n - x_n^{d_0-1} & \cdots & y_n^{\ell-2}(y_n/x_n - x_n^{d_{\ell-1}-1}) \end{pmatrix}.$$

and

$$(12) \quad B := \underbrace{\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \end{pmatrix}}_{d_0} \underbrace{\begin{pmatrix} & & & & & & & & \cdots & & & & & \\ & & & & & & & & \ddots & & & & & \\ & & & & & & & & & & & & & & \\ & & & & & & & & & & & & & & \\ & & & & & & & & & & & & & & \end{pmatrix}}_{d_1} \underbrace{\begin{pmatrix} & & & & & & & & & & & & & & \\ & & & & & & & & & & & & & & & \\ & & & & & & & & & & & & & & & \\ & & & & & & & & & & & & & & & \\ & & & & & & & & & & & & & & & \end{pmatrix}}_{d_\ell}.$$

Then, one easily proves that

$$DV - VZ = GB,$$

where  $Z$  is the upper shift matrix of format  $n+1$  defined in (7). Hence,  $V$  has displacement rank at most  $\ell+1$  with respect to  $\nabla_{D,Z}$ , and we can use Algorithm 3.3 to compute a nonzero element in the kernel of  $V$ .

**ALGORITHM 4.1.** *On input  $(x_1, y_1), \dots, (x_n, y_n) \in \mathbb{F}_q^2$  and integers  $d_0, d_1, \dots, d_\ell$  such that  $\sum_{i=0}^\ell d_i = n + 1$  and such that none of the  $x_i$  is zero, this algorithm computes a polynomial  $F(x, y) = \sum_{i=0}^\ell F_i(x)y^i$  such that  $F(x_i, y_i) = 0$  for  $i = 1, \dots, n$ .*

- (1) *Run Algorithm 3.3 on input  $D, G, B$ , where  $D$  is the diagonal matrix with entries  $1/x_1, \dots, 1/x_n$ , and  $G, B$  given in (11) and (12), respectively. Let  $v = (v_{00}, v_{01}, \dots, v_{0,d_0-1}, \dots, v_{\ell,0}, v_{\ell,1}, \dots, v_{\ell,d_\ell-1})$  denote the output.*
- (2) *Output  $F(x, y) = \sum_{i=0}^\ell F_i(x)y^i$  where  $F_i(x) = v_{i,0} + v_{i,1}x + \cdots + v_{i,d_i-1}x^{d_i-1}$ .*

Theorem 3.4 immediately implies the following result.

**THEOREM 4.2.** *Algorithm 4.1 correctly computes its output with  $O(n^2\ell)$  operations over the field  $K$ .*

There are various methods to extend the algorithm to the case where one of the  $x_i$  is zero. We will sketch one of them. Without loss of generality, assume that  $x_1 = 0$ . In this case, deletion of the first row of the matrix  $V$  yields another matrix  $\tilde{V}$  of the form given in (10) in which none of the  $x_i$  is zero. The matrix  $V$  has then the following displacement structure

$$\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1/x_2 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1/x_n \end{pmatrix} V - VZ = \begin{pmatrix} 0_{1 \times (\ell+1)} & 1 \\ G & 0_{(n-1) \times 1} \end{pmatrix} \cdot \begin{pmatrix} B \\ x \end{pmatrix},$$

where  $x$  is the vector

$$x = \left( \underbrace{1, -1, 0, \dots, 0}_{d_0}, \underbrace{y_1, -y_1, 0, \dots, 0}_{d_1}, \dots, \underbrace{y_\ell^\ell, -y_1^\ell, 0, \dots, 0}_{d_\ell} \right),$$

$G$  is the matrix obtained from the one given in (11) by deleting the first row, and  $0_{a \times b}$  denotes the  $a \times b$ -matrix consisting of zeros. We can now run Algorithm 3.3 on the new set of matrices to obtain a nonzero element in the kernel of  $V$ .

We finish this section with an example. Suppose that  $C$  is the Reed-Solomon code of dimension 4 given as the set of evaluations of polynomials of degree at most 3 over the field  $\mathbb{F}_{31}$  at the points  $x_1 = 1, x_2 = 2, \dots, x_{30} = 30$ . This code can correct up to 15 errors with lists of size at most 2. In this case,  $\ell = 2$ ,  $n = 30$ ,  $k = 4$ ,  $b = \lfloor n/(\ell + 1) + \ell k/2 + 1 \rfloor = 14$ ,  $d_0 = 14$ ,  $d_1 = 11$ , and  $d_2 = 8$ .

Suppose that we have received the vector

$$y = (3, 13, 0, 6, 7, 24, 19, 25, 1, 17, 19, 5, 10, 0, 19, 2, 4, 23, 28, 23, 29, 7, 8, 12, 27, 24, 15, 6, 22, 30)$$

Then the matrix  $G$  given in (11) is as follows  $G^\top =$

$$\left( \begin{array}{cccccccccccccccccccccc} 1 & 16 & 21 & 8 & 25 & 26 & 9 & 4 & 7 & 28 & 17 & 13 & 12 & 20 & 29 & 2 & 11 & 19 & 18 & 14 & 3 & 24 & 27 & 22 & 5 & 6 & 23 & 10 & 15 & 30 \\ 2 & 14 & 7 & 15 & 15 & 29 & 28 & 22 & 20 & 2 & 23 & 17 & 16 & 3 & 28 & 0 & 10 & 14 & 25 & 2 & 3 & 0 & 15 & 4 & 17 & 25 & 6 & 22 & 28 & 2 \\ 6 & 25 & 0 & 3 & 12 & 23 & 15 & 26 & 2 & 9 & 28 & 14 & 3 & 0 & 3 & 6 & 14 & 16 & 20 & 6 & 22 & 25 & 15 & 16 & 25 & 19 & 14 & 24 & 15 & 0 \end{array} \right)$$

The matrix  $B$  in (12) is as follows:

The matrix  $D$  of Algorithm 4.1 is the diagonal matrix with diagonal entries

$$(1, 16, 21, 8, 25, 26, 9, 4, 7, 28, 17, 13, 12, 20, 29, 2, 11, 19, 18, 14, 3, 24, 27, 22, 5, 6, 23, 10, 15, 30).$$

In the first round of Algorithm 3.3, the first column computed in step 2(a) is

The first row, computed at step 2(c), is

The loop in step 2 of Algorithm 3.3 is performed 29 times. The kernel element computed at the end equals

$$(12, 5, 4, 9, 20, 19, 28, 26, 23, 18, 11, 16, 14, 25, 26, 15, 14, 25, 1, 25, 29, 27, 9, 17, 6, 24, 30, 7, 20, 1, 0).$$

It corresponds to the bivariate polynomial  $F(x, y) =$

$$(12 + 5x + 4x^2 + 9x^3 + 20x^4 + 19x^5 + 28x^6 + 26x^7 + 23x^8 + 18x^9 + 11x^{10} + 16x^{11} + 14x^{12} + 25x^{13}) + \\ (26 + 15x + 14x^2 + 25x^3 + x^4 + 25x^5 + 29x^6 + 27x^7 + 9x^8 + 17x^9 + 6x^{10})y + \\ (24 + 30x + 7x^2 + 20x^3 + x^4)y^2.$$

Applying the algorithm of, e.g., [11], we obtain

$$F(x, y) \equiv (y - (1+x+x^3))((y-16)+(9y-1)x+(6-y)x^2+(29+6y)x^3+(22y+28)x^4) \bmod x^5.$$

which shows that there is at most one polynomial,  $f(x) = 1+x+x^3$ , which satisfies  $F(x, f(x)) = 0$ . A further calculation shows that this is indeed the case. Hence, there is only one codeword in  $C$  of distance at most 15 from the received word  $y$ , namely the codeword corresponding to  $f$ .

## 5. The Algorithm of Shokrollahi-Wasserman

In [44] the authors generalize Sudan's algorithm to algebraic-geometric (AG-) codes. We briefly discuss this generalization. Let  $\mathcal{X}$  be an irreducible algebraic curve over the finite field  $\mathbb{F}_q$ , let  $Q, P_1, \dots, P_n$  be distinct  $\mathbb{F}_q$ -rational points of  $\mathcal{X}$ , and let  $L(\alpha Q)$  denote the linear space of the divisor  $\alpha Q$ , i.e., the space generated by all functions in the function field of  $\mathcal{X}$  that have only a pole of order at most  $\alpha$  at  $Q$ . The (one-point) AG-code associated to these data is then defined as the image of the  $\mathbb{F}_q$ -linear map  $L(\alpha Q) \rightarrow \mathbb{F}_q^n$  mapping a function  $f$  to the vector  $(f(P_1), \dots, f(P_n))$ . It is well known that this linear code has dimension  $k \geq \alpha - g + 1$  and minimum distance  $d \geq n - \alpha$ , where  $g$  denotes the genus of the curve. Suppose that we want to decode this code with a list of length at most  $\ell$ . Let  $\beta := \lfloor (n+1)/(\ell+1) + \ell\alpha/2 + g \rfloor$ . Let  $\varphi_1, \dots, \varphi_t$ ,  $t = \beta - g + 1$ , be elements of  $L(\beta Q)$  with strictly increasing pole orders at  $Q$ . Let  $(y_1, \dots, y_n)$  be the received word. The algorithm in [44] first finds functions  $u_0, \dots, u_\ell$  with  $u_i \in L((\beta - i\alpha)Q)$  such that  $\sum_j u_i(P_j)y_j^i = 0$  for all  $1 \leq j \leq n$ . This step is accomplished by computing a nonzero element in the kernel of the matrix

$$(13) \quad V := \begin{pmatrix} V_0 & V_1 & \dots & V_\ell \end{pmatrix}$$

with

$$V_0 = \begin{pmatrix} \varphi_1(P_1) & \dots & \varphi_{s_0}(P_1) \\ \varphi_1(P_2) & \dots & \varphi_{s_0}(P_2) \\ \vdots & \ddots & \vdots \\ \varphi_1(P_n) & \dots & \varphi_{s_0}(P_n) \end{pmatrix}, \\ V_1 = \begin{pmatrix} y_1\varphi_1(P_1) & \dots & y_1\varphi_{s_1}(P_1) \\ y_2\varphi_1(P_2) & \dots & y_2\varphi_{s_1}(P_2) \\ \vdots & \ddots & \vdots \\ y_n\varphi_1(P_n) & \dots & y_n\varphi_{s_1}(P_n) \end{pmatrix}, \quad V_\ell = \begin{pmatrix} y_1^\ell\varphi_1(P_1) & \dots & y_1^\ell\varphi_{s_\ell}(P_1) \\ y_2^\ell\varphi_1(P_2) & \dots & y_2^\ell\varphi_{s_\ell}(P_2) \\ \vdots & \ddots & \vdots \\ y_n^\ell\varphi_1(P_n) & \dots & y_n^\ell\varphi_{s_\ell}(P_n) \end{pmatrix}.$$

where  $s_j = \beta - j\alpha - g + 1$  for  $j < \ell$  and  $\sum_{j=0}^\ell (s_j + 1) = n + 1$ . To simplify the discussions, we assume that there are two functions  $\varphi, \psi \in L(3Q)$  such that all the  $\varphi_i$  are of the form  $\varphi^a \psi^b$  and such that the order of poles at  $Q$  of  $\varphi$  is smaller than that of  $\psi$ . Further, we assume that  $\varphi$  does not vanish at any of the points  $P_1, \dots, P_n$ . We remark that the method described below can be modified to deal with a situation in which any of the above assumptions is not valid.

Let  $d$  be the order of poles of  $\varphi$  at  $Q$ . Then it is easily seen that any element of  $L(\beta Q)$  is of the form  $\varphi^a \psi^b$ , where  $0 \leq b < d$ . Now we divide each of the blocks of the matrix  $V$  into subblocks in the following way: by changing the order of the columns, we write the  $t$ -th block of  $V$  in the form

$$V_t = \begin{pmatrix} V_{t,0} & \dots & V_{t,s} \end{pmatrix}$$

with

$$V_{t,0} = \begin{pmatrix} \varphi(P_1)^0 \psi(P_1)^0 y_1^t & \dots & \varphi(P_1)^{a_0} \psi(P_1)^0 y_1^t \\ \varphi(P_2)^0 \psi(P_2)^0 y_2^t & \dots & \varphi(P_2)^{a_0} \psi(P_2)^0 y_2^t \\ \vdots & \ddots & \vdots \\ \varphi(P_n)^0 \psi(P_n)^0 y_n^t & \dots & \varphi(P_n)^{a_0} \psi(P_n)^0 y_n^t \end{pmatrix}$$

and

$$V_{t,s} = \begin{pmatrix} \varphi(P_1)^0 \psi(P_1)^s y_1^t & \dots & \varphi(P_1)^{a_s} \psi(P_1)^s y_1^t \\ \varphi(P_2)^0 \psi(P_2)^s y_2^t & \dots & \varphi(P_2)^{a_s} \psi(P_2)^s y_2^t \\ \vdots & \ddots & \vdots \\ \varphi(P_n)^0 \psi(P_n)^s y_n^t & \dots & \varphi(P_n)^{a_s} \psi(P_n)^s y_n^t \end{pmatrix}$$

By the above remarks,  $s < d$ . Let now  $D$  be the diagonal matrix  $\text{diag}[\varphi]_{1,n}$ , and let  $Z$  be the upper shift matrix of format  $n+1$ . Then in a similar way as in the last section one can construct a matrix  $G \in \mathbb{F}_q^{n \times d\ell}$  and a matrix  $B \in \mathbb{F}_q^{d\ell \times m}$  such that  $DV - VZ = GB$ , where  $Z$  is the upper shift matrix of format  $n+1$ . Using Algorithm 3.3 we then obtain the following result.

**THEOREM 5.1.** *Using Algorithm 3.3, one can compute a nonzero element in the kernel of the matrix  $V$  of (13) with  $O(n^2 d\ell)$  operations over the base field.*

A final estimate of the running time of this algorithm has to take into account the relationship between  $d$  and  $n$ . This depends on the specific structure of the curve and the divisor used in the definition. For instance, if the curve has a nonsingular plane model of the form  $G(x, y) = 0$ , and  $G$  is a monic polynomial of degree  $d$  in the variable  $y$ , and if we choose for  $Q$  the common pole of the functions  $x$  and  $y$ , then the above conditions are all valid (see also [20]). As an example, consider the case of a Hermitian curve defined over  $\mathbb{F}_{q^2}$  by the equation  $x^{q+1} = y^q + y$ . In this case, the parameter  $d$  equals  $q+1$ , which  $n = q^3$ . As a result, the algorithm uses  $O(n^{7/3}\ell)$   $\mathbb{F}_q$ -operations. Note that for  $\ell = 1$ , i.e., for unique decoding, this is exactly the running time of the algorithm presented in [20].

We remark that there are plenty of other fast algorithms for (conventional) decoding of AG-codes which make use of the structure of the matrices, though in a more implicit form than given here, see, e.g., [19] and the references therein, or the work [28].

## 6. The Algorithm of Guruswami-Sudan

In [15] the authors describe an extension of algorithms presented in [45] and [44] in which they use a variant of Hermite interpolation rather than a Lagrange interpolation. In the case of Reed-Solomon codes, the input to the algorithm is a set of  $n$  points  $(x_i, y_i)$  in the affine plane over  $\mathbb{F}_q$ , and parameters  $r, k$ , and  $\ell$ . Let  $\beta$  be the smallest integer such that  $(\ell+1)\beta > \binom{\ell+1}{2}k + \binom{r+1}{2}n$ . The output of the algorithm is a nonzero bivariate polynomial  $G = \sum_{i=0}^{\ell} G_i(x)y^i$  such that  $\deg(G_i) < \beta - i(k-1)$ ,  $G(x_i, y_i) = 0$  for all  $i \leq n$ , and  $G(x+x_i, y+y_i)$  does not contain any monomial of

degree less than  $r$ . Such a polynomial  $G$  corresponds to a nonzero element in the kernel of a certain matrix  $V$  which we will describe below. Let  $t \leq n$  be a fixed positive integer. For  $0 \leq i < r$  and  $0 \leq j \leq \ell$  let  $V_{ij}^t \in \mathbb{F}_q^{(r-i) \times (\beta-j(k-1))}$  be the matrix having  $(\mu, \nu)$ -entry equal to  $\binom{\nu}{\mu} x_t^{\nu-\mu}$ , where  $0 \leq \mu < r-i$  and  $0 \leq \nu < \beta-j(k-1)$ . Now define the matrix  $V_t$  as a block matrix with  $r$  block rows and  $\ell+1$  block columns, with block entry  $(i, j)$  equal to  $\binom{j}{i} y_t^{j-i} V_{ij}^t$ . The matrix  $V$  then equals

$$(14) \quad V = \begin{pmatrix} V_1 \\ V_2 \\ \vdots \\ V_n \end{pmatrix}.$$

$V$  has  $m := \binom{r+1}{2}n$  rows and  $s := (\ell+1)\beta - \binom{\ell+1}{2}(k-1)$  columns (and  $s > m$ ). In the following we will show that  $V$  has a displacement structure with respect to suitable matrices.

Let  $J_i^t$  denote the  $i \times i$ -Jordan block having  $x_t$  in its main diagonal and 1's on its lower sub-diagonal. Let  $J^t$  be the block diagonal matrix with block diagonal entries  $J_r^t, \dots, J_1^t$ . Let  $J$  be the block diagonal matrix with block diagonal entries  $J^1, \dots, J^n$ :

$$(15) \quad J := \underbrace{\begin{pmatrix} J^1 & 0 & \cdots & 0 \\ 0 & J^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & J^n \end{pmatrix}}_{n \binom{r+1}{2}}$$

with

$$J^t := \underbrace{\begin{pmatrix} J_r^t & 0 & \cdots & 0 \\ 0 & J_{r-1}^t & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & J_1^t \end{pmatrix}}_{\binom{r+1}{2}}, \quad J_i^t := \underbrace{\begin{pmatrix} x_t & 0 & 0 & \cdots & 0 & 0 \\ 1 & x_t & 0 & \cdots & 0 & 0 \\ 0 & 1 & x_t & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & x_t \end{pmatrix}}_i.$$

Then, a quick calculation shows that

$$(16) \quad JV - VZ_s^\top = HU$$

where  $Z_s$  is the upper shift matrix of format  $s$ , and  $H$  and  $U$  are described as follows: for  $1 \leq t \leq n$  let  $H_t = \in \mathbb{F}_q^{(\frac{r+1}{2}) \times (\ell+1)}$  be given by  

$$(17)$$

$$(17) \quad \begin{pmatrix} b_{0,0}^{d_0,0} - b_{1,0}^{0,0} & b_{1,0}^{d_1,0} - b_{2,0}^{0,0} & \cdots & b_{\ell-1,0}^{d_{\ell-1},0} - b_{\ell,0}^{0,0} & b_{\ell,0}^{d_\ell,0} \\ b_{0,0}^{d_0,1} - b_{1,0}^{0,1} & b_{1,0}^{d_1,1} - b_{2,0}^{0,1} & \cdots & b_{\ell-1,0}^{d_{\ell-1},1} - b_{\ell,0}^{0,1} & b_{\ell,0}^{d_\ell,1} \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ b_{0,0}^{d_0,r-1} - b_{1,0}^{0,r-1} & b_{1,0}^{d_1,r-1} - b_{2,0}^{0,r-1} & \cdots & b_{\ell-1,0}^{d_{\ell-1},r-1} - b_{\ell,0}^{0,r-1} & b_{\ell,0}^{d_\ell,r-1} \end{pmatrix}$$


---


$$H_t := \begin{pmatrix} b_{0,1}^{d_0,0} - b_{1,1}^{0,0} & b_{1,1}^{d_1,0} - b_{2,1}^{0,0} & \cdots & b_{\ell-1,1}^{d_{\ell-1},0} - b_{\ell,1}^{0,0} & b_{\ell,1}^{d_\ell,0} \\ b_{0,1}^{d_0,1} - b_{1,1}^{0,1} & b_{1,1}^{d_1,1} - b_{2,1}^{0,1} & \cdots & b_{\ell-1,1}^{d_{\ell-1},1} - b_{\ell,1}^{0,1} & b_{\ell,1}^{d_\ell,1} \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ b_{0,1}^{d_0,r-2} - b_{1,1}^{0,r-2} & b_{1,1}^{d_1,r-2} - b_{2,1}^{0,r-2} & \cdots & b_{\ell-1,1}^{d_{\ell-1},r-2} - b_{\ell,1}^{0,r-2} & b_{\ell,1}^{d_\ell,r-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ b_{0,r-11}^{d_0,0} - b_{1,r-11}^{0,0} & b_{1,r-11}^{d_1,0} - b_{2,r-11}^{0,0} & \cdots & b_{\ell-1,r-11}^{d_{\ell-1},0} - b_{\ell,r-11}^{0,0} & b_{\ell,r-11}^{d_\ell,0} \end{pmatrix}$$

where  $b_{c,d}^{e,f} := \binom{c}{d} \binom{e}{f} y_t^{c-d} x_t^{e-f}$ . Then

$$(18) \quad H = \begin{pmatrix} H_1 \\ H_2 \\ \vdots \\ H_n \end{pmatrix},$$

and

$$(19) \quad U := \left( \begin{array}{cccccc|cccccc|c} 0 & 0 & 0 & \cdots & 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 \end{array} \right).$$

Unfortunately, we cannot apply the results of Section 2 to this situation directly, since  $J$  is not a diagonal matrix and  $Z_s^\top$  is not upper triangular. A remedy of the situation is as follows. Let  $\mathbb{F}$  be a suitable extension of  $\mathbb{F}_q$  which contains  $s$  nonzero pairwise distinct elements  $\epsilon_1, \dots, \epsilon_s$ . Let  $W$  be the  $s \times s$ -Vandermonde

matrix having  $(i, j)$ -entry  $\epsilon_i^{j-1}$  and let  $\Delta$  be the diagonal matrix with diagonal entries  $1/\epsilon_1, \dots, 1/\epsilon_s$ . Then we have the following displacement structure for  $W$

$$(20) \quad \Delta W - WZ = EF.$$

where  $E = (1/\epsilon_1, 1/\epsilon_2, \dots, 1/\epsilon_s)^\top$  and  $F = (1, 0, \dots, 0)$ . Transposing both sides of (16) we obtain the equation  $Z_s V^\top - V^\top J^\top = -F^\top E^\top$ . Multiplying both sides of this equation with  $W$  from the left, multiplying both sides of (20) with  $V^\top$  from the right, and adding the equations gives the displacement equation for the matrix  $WV^\top$ :

$$(21) \quad \Delta WV^\top - WV^\top J^\top = EFV^\top - WU^\top H^\top = (E \mid -WU^\top) \cdot \begin{pmatrix} FV^\top \\ H^\top \end{pmatrix} =: GB.$$

Written out explicitly, we have

$$(22) \quad G = \begin{pmatrix} \epsilon_1^{D_0-1} & -\epsilon_1^{D_1-1} & -\epsilon_1^{D_2-1} & \dots & -\epsilon_1^{D_\ell-1} \\ \epsilon_2^{D_0-1} & -\epsilon_2^{D_1-1} & -\epsilon_2^{D_2-1} & \dots & -\epsilon_2^{D_\ell-1} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ \epsilon_s^{D_0-1} & -\epsilon_s^{D_1-1} & -\epsilon_s^{D_2-1} & \dots & -\epsilon_s^{D_\ell-1} \end{pmatrix},$$

$$(23) \quad B = \left( \underbrace{\begin{matrix} 1 & 0 & \dots & 0 \end{matrix}}_{H_1^\top} \quad \underbrace{\begin{matrix} 1 & 0 & \dots & 0 \end{matrix}}_{H_2^\top} \quad \dots \quad \underbrace{\begin{matrix} 1 & 0 & \dots & 0 \end{matrix}}_{H_n^\top} \right),$$

where  $D_0 := 0$ , and  $D_i := d_0 + \dots + d_{i-1}$  for  $i > 0$ , and  $H_i$  as defined in (17).

The matrix  $WV^\top$  has the right displacement structure, since  $\Delta$  is diagonal and  $J^\top$  is upper triangular. However, since we are interested in an element in the right kernel of  $V$ , we need to compute an element in the *left* kernel of  $WV^\top$ . This computation can be done using similar techniques as above by considering an appropriate displacement structure for the matrix  $(WV^\top \mid I)$ . However, certain complications arise in this case due to the fact that the displacement operator for this matrix which maintains a low displacement rank is not an isomorphism. Therefore, we defer the design of that algorithm to the appendix. Using that procedure, we obtain the following overall algorithm.

**ALGORITHM 6.1.** *On input  $(x_1, y_1), \dots, (x_n, y_n) \in \mathbb{F}_q^2$  where none of the  $x_i$  is zero, a positive integer  $r$ , and integers  $d_0, d_1, \dots, d_\ell$  such that  $\sum_{i=0}^\ell (d_i + 1) =: s > \binom{r+1}{2}n$ , this algorithm computes a polynomial  $F(x, y) = \sum_{i=0}^\ell F_i(x)y^i$  such that  $\deg(F_i) < d_i$ ,  $F(x_t, y_t) = 0$  for  $t = 1, \dots, n$ , and such that  $F(x + x_t, y + y_t)$  does not have any monomial of degree less than  $r$ .*

- (1) Let  $d := \lceil \log_q(s+n+1) \rceil$  and construct the finite field  $\mathbb{F}_{q^d}$ .
- (2) Choose  $s$  pairwise distinct nonzero elements  $\epsilon_1, \dots, \epsilon_s$  in  $\mathbb{F}_{q^d}$  such that  $\epsilon_i \neq x_j$  for  $i \neq j$ .
- (3) Run Algorithm A.1 on input

$$D, J, G, B, \underbrace{(1, 1, \dots, 1)}_{s \text{ times}},$$

where  $D$  is the diagonal matrix with diagonal entries  $1/\epsilon_1, \dots, 1/\epsilon_s$ ,  $J$  is the matrix defined in (15),  $G$  is the matrix defined in (22), and  $B$  is the matrix defined in (23). Let  $w$  denote the output.

- (4) Compute  $v := w \cdot W$  where  $W$  is the Vandermonde matrix with  $(i, j)$ -entry equal to  $\epsilon_i^{j-1}$ . Let  $v = (v_{00}, v_{01}, \dots, v_{0,d_0-1}, \dots, v_{\ell,0}, v_{\ell,1}, \dots, v_{\ell,d_\ell-1})$  denote the output.
- (5) Output  $F(x, y) = \sum_{i=0}^{\ell} F_i(x) y^i$  where  $F_i(x) = v_{i,0} + v_{i,1}x + \dots + v_{i,d_i-1}x^{d_i-1}$ .

**THEOREM 6.2.** *Algorithm 6.1 correctly computes its output with  $O(s^2\ell)$  operations over  $\mathbb{F}_{q^d}$ , or, equivalently,  $O(s^2\ell \log_q(s)^2)$  operations over  $\mathbb{F}_q$ .*

**PROOF.** The finite field  $\mathbb{F}_{q^d}$  can be constructed with a randomized algorithm with an expected number of  $O(d^3 \log(d) \log(dq))$  operations over the field  $\mathbb{F}_q$  [12, Th. 14.42]. This cost is negligible compared to the cost of the subsequent steps. As proved in the appendix, Step (3) requires  $O(\ell s^2)$  operations over  $\mathbb{F}_{q^d}$ . Steps (4), (5), and (6) each require  $O(s^2)$  operations over  $\mathbb{F}_{q^d}$ . Each  $\mathbb{F}_{q^d}$ -operation can be simulated with  $O(d^2)$  operations over  $\mathbb{F}_q$ . The result follows.  $\square$

## 7. Parallel Algorithms

The aim of this section is to show how the displacement method can be used to obtain parallel algorithms for computing a nontrivial element in the kernel of a structured matrix. The model of parallel computation that we use is a PRAM (Parallel Random Access Machine). A PRAM consists of several independent sequential processors, each with its own private memory, communicating with one another through a global memory. In one unit of time, each processor can read one global or local memory location, execute a single random access operation, and write into one global or local memory location. We assume in the following the each processor in the PRAM is capable of performing operations over  $\mathbb{F}_q$  in a single step. The *running time* of a PRAM is the number of steps it performs to finish its computation.

We will use the following standard facts: one can multiply  $m \times \alpha$ - with  $\alpha \times k$ -matrices on a PRAM in  $O(\alpha k)$  time on  $m$  processors. This is because each of the  $m$  processors performs independently a vector-matrix multiplication of size  $1 \times \alpha$  and  $\alpha \times k$ . Further, one can solve an upper triangular  $m \times m$ -system of equations in time  $O(m)$  on  $m$  processors. There are various algorithms for this problem. One is given in [29, Sect. 2.4.3]. As a result, it is easily seen that Algorithm 2.3 can be customized to run in parallel, if steps (1) and (3) can be performed in parallel. More precisely, if these steps can be performed on  $m$  processors in time  $O(\alpha)$ , then the whole algorithm can be customized to run in time  $O(\alpha(m+n))$  on  $m$  processors.

However, one cannot always perform steps (1) and (3) in parallel. Whether or not this is possible depends heavily on the displacement structure used. For instance, it is easy to see that the recovery algorithm given in Section 3 cannot be parallelized (at least in an obvious way). The reason for this is that for obtaining the  $i$ th entry of the first row one needs to know the value of the  $(i-1)$ st entry. To obtain the  $n$ th entry one thus needs to know the value of all the previous entries. It is therefore not possible to perform this step in time  $O(\alpha)$  even if the number of processors is unbounded!

A closer look at this problem reveals the following: if  $V$  has a displacement structure of the type  $DV - V\Delta = GB$  with *diagonal* matrices  $D$  and  $\Delta$ , then one can recover the first row and the first column of  $V$  in parallel, if the nonzero entries of  $D$  and  $\Delta$  are pairwise distinct. The algorithm for this task is rather simple and is given below.

The problem with this approach is that the matrices  $V$  we are studying do not necessarily have low displacement structure with respect to a pair of diagonal matrices. This problem can be fixed in certain cases. For instance, suppose that  $V \in \mathbb{F}_q^{m \times n}$  has low displacement rank with respect to  $D$  and  $Z_n$ , where  $D$  is diagonal and  $Z_n$  is the upper shift matrix of format  $n$ :

$$(24) \quad DV - VZ_n = G_1 B_1.$$

Let  $W$  be a Vandermonde matrix whose  $(i, j)$ -entry is  $\epsilon_i^{j-1}$ , where  $\epsilon_1, \dots, \epsilon_n$  are elements in  $\mathbb{F}_q$  that are pairwise distinct and different from the diagonal entries of  $D$ . Then we have the following displacement structure for  $W$

$$\Delta W - WZ_n^\top = G_2 B_2$$

where  $G_2 = (\epsilon_1^n, \dots, \epsilon_n^n)^\top$  and  $B_2 = (0, 0, \dots, 0, 1)$ . Transposing this equation and multiplying with  $V$  from the left gives

$$(25) \quad VZ_n W^\top - W^\top \Delta = (VB_2^\top) G_2.$$

Multiplying (24) with  $W$  from the right, and adding that to the previous equation gives

$$(26) \quad DVW^\top - VW^\top \Delta = (G_1 | VB_2^\top) \begin{pmatrix} B_1 W^\top \\ G_2 \end{pmatrix} =: GB.$$

This is now the correct displacement structure, but for the wrong matrix  $VW^\top$ . However, if  $w$  is a nonzero vector in the kernel of  $VW^\top$ , then  $v := W^\top w$  is in the kernel of  $V$ . Moreover, since  $W$  is invertible,  $v$  is nonzero, hence is the desired solution.

The rest of this section deals with putting these ideas into effective use. We will first start by designing an algorithm that computes the first row and the first column of the matrix  $V$  that has a displacement structure with respect to two diagonal matrices.

**ALGORITHM 7.1.** *The input to the algorithm are two diagonal matrices  $D$  and  $\Delta$ , and two additional matrices  $G = (G_{ij}) \in \mathbb{F}_q^{m \times \alpha}$  and  $B = (B_{ij}) \in \mathbb{F}_q^{\alpha \times n}$ . We assume that none of the diagonal entries  $x_1, \dots, x_n$  is equal to any of the diagonal entries  $z_1, \dots, z_m$  of  $\Delta$ . The output of the algorithm are the first row  $(r_1, \dots, r_m)$  and the first column  $(c_1, \dots, c_n)^\top$  of the matrix  $V$  given by the displacement structure  $DV - V\Delta = GB$ .*

- (1) Compute the first row  $(\rho_1, \dots, \rho_m)$  and the first column  $(\gamma_1, \dots, \gamma_n)^\top$  of  $GB$ .
- (2) For  $i = 1, \dots, n$  set in parallel  $c_i := \gamma_i / (x_i - z_1)$ .
- (3) For  $i = 1, \dots, m$  set in parallel  $r_i := \rho_i / (x_1 - z_i)$ .

**PROPOSITION 7.2.** *The above algorithm correctly computes its output with  $O(\alpha)$  operations on a PRAM with  $\max\{m, n\}$  processors.*

**PROOF.** Correctness of the algorithm is easily obtained the same way as that of Algorithm 3.1. As for the running time, note that Step (1) can be performed in time  $O(\alpha)$  on  $\max\{m, n\}$  processors. Steps (2) and (3) obviously need a constant number of steps per processor.  $\square$

To come up with a memory-efficient procedure, we will customize Algorithm 2.5 rather than Algorithm 2.3. For this we need to have a displacement structure for

$\binom{V}{I_n}$ . This is given by the following:

$$(27) \quad \begin{pmatrix} D & 0 \\ 0 & \Delta \end{pmatrix} \begin{pmatrix} V \\ I_n \end{pmatrix} - \begin{pmatrix} V \\ I_n \end{pmatrix} \Delta = \begin{pmatrix} G \\ 0 \end{pmatrix} B.$$

ALGORITHM 7.3. On input a diagonal matrix  $D \in \mathbb{F}_q^{n \times n}$  with diagonal entries  $x_1, \dots, x_n$ , a diagonal matrix  $\Delta \in \mathbb{F}_q^{(n+1) \times (n+1)}$  with diagonal entries  $z_1, \dots, z_{n+1}$  such that the  $x_i$  and the  $z_j$  are pairwise distinct, and  $z_i \neq x_j$  for  $i \neq j$ , a matrix  $G \in \mathbb{F}_q^{n \times \alpha}$ , and a matrix  $B \in \mathbb{F}_q^{\alpha \times (n+1)}$ , the algorithm computes a nonzero vector  $v \in \mathbb{F}_q^n$  in the kernel of the matrix  $V$  given by the displacement equation  $DV - V\Delta = GB$ .

(1) Set  $G_0 := \begin{pmatrix} G \\ 0 \end{pmatrix}$ ,  $B_0 := B$ .

(2) For  $i = 0, \dots, n-1$  do:

(a) Let  $D_i$  be the diagonal matrix with diagonal entries  $x_{i+1}, \dots, x_n, z_1, \dots, z_i$ , and let  $\Delta_i$  be the diagonal matrix with diagonal entries  $z_{i+1}, \dots, z_{n+1}$ .

(a) Use Algorithm 7.1 to compute the first column  $(c_1, c_2, \dots, c_{2n+1-i})^\top$  of the matrix  $V_i$  given by the displacement equation

$$\begin{pmatrix} D_i & 0 \\ 0 & \Delta \end{pmatrix} V_i - V_i \Delta_i = G_i B_i.$$

(b) If  $c_1 = \dots = c_{n-i} = 0$ , then **output** the vector

$v := (c_{n-i+1}, \dots, c_n, 1, 0, \dots, 0)$ . and **Stop**. Otherwise, find the smallest  $k$  such that  $c_k \neq 0$ , interchange  $x_1$  and  $x_k$ ,  $c_1$  and  $c_k$ , and the first and the  $k$ th rows of  $G_i$ .

(c) Use Algorithm 7.1 again to compute in parallel the first row

$(r_1, \dots, r_{n-i+1})$  of the matrix  $V_i$  given by  $D_i V_i - V_i \Delta_i = G_i B_i$  (note that  $V_i$  is not necessarily the same as in Step (2a) since  $D_i$  and  $G_i$  may not be the same).

(d) For  $j = 2, \dots, 2n-i+1$  replace row  $j$  of  $G_i$  by  $-c_j/c_1$  times the first row plus the  $j$ -th row. Set  $G_{i+1}$  as the matrix formed from  $G_i$  by deleting the first row.

(e) For  $j = 2, \dots, n-i+1$  replace the  $j$ -th column of  $B_i$  by  $-r_j/r_1$  times the first column plus the  $j$ -th column. Set  $B_{i+1}$  as the matrix formed from  $B_i$  by deleting the first column.

THEOREM 7.4. The above algorithm correctly computes its output and uses  $O(\alpha n)$  operations on a PRAM with  $n+1$  processors.

PROOF. This algorithm is a customized version of Algorithm 2.3 and its correctness follows from that of the latter. As for the running time, note that Step (2a) takes  $O(\alpha)$  steps on a PRAM with  $2n+1-i$  processors by Proposition 7.2 ( $G_i$  has  $2n+1-i$  rows). However, a simple induction shows that the last  $n+1-i$  rows of  $G_i$  are zero. As a result, we only need  $n-i$  processors for this step. Step (2c) can be performed in time  $O(\alpha)$  on  $n+1-i$  processors by Proposition 7.2. Steps (2d) and (2e) can be performed in time  $O(\alpha)$  on  $n+1-i$  processors (again, the number of processors is not equal to the number of rows of  $G_i$  since the last  $n+1-i$  rows of  $G_i$  are zero). At each round of the algorithm the processors can be reused, so the total number of processors equals to the maximum number at each round, i.e.,

it equals  $n + 1$ . The running time is at most  $O(n\alpha)$  since step (2) is performed at most  $n$  times.  $\square$

The above algorithm can now be used as a major building block for solving the 2-dimensional interpolation problem in parallel.

**ALGORITHM 7.5.** *On input  $(x_1, y_1), \dots, (x_n, y_n) \in \mathbb{F}_q^2$  and integers  $d_0, d_1, \dots, d_\ell$  such that none of the  $x_i$  is zero and such that  $\sum_{i=0}^\ell d_i = n + 1$ , this algorithm computes a polynomial  $F(x, y) = \sum_{i=0}^\ell F_i(x)y^i$  such that  $\deg(F_i) < d_i$  for  $i = 0, \dots, \ell$  and  $F(x_t, y_t) = 0$  for  $t = 1, \dots, n$ .*

- (1) Let  $d := \lceil \log_q(2n + 1) \rceil$  and construct the field  $\mathbb{F}_{q^d}$ . Find  $n + 1$  elements  $\epsilon_1, \dots, \epsilon_{n+1}$  in  $\mathbb{F}_{q^d}$  that are pairwise distinct and such that  $\epsilon_i \neq 1/x_j$  for  $i \neq j$ .
- (2) Set  $G := (G_1 \mid v_1)$  where  $G_1$  is the matrix given in (11) and  $v_1$  is the last column of the matrix  $V$  given in (10).
- (3) Compute  $\tilde{B} := B_1 W^\top$  where  $B$  is the matrix given in (12) and  $W$  is the  $(n + 1) \times (n + 1)$  Vandermonde matrix having  $(i, j)$ -entry  $\epsilon_i^{j-1}$ . Set  $B := \begin{pmatrix} \tilde{B} \\ G_2 \end{pmatrix}$  where  $G_2$  is the vector  $(\epsilon_1^{n+1}, \dots, \epsilon_{n+1}^{n+1})$ .
- (4) Run Algorithm 7.3 on input  $D, \Delta, G, B$ , where  $D$  is a diagonal matrix with diagonal entries  $1/x_1, \dots, 1/x_n$ ,  $\Delta$  is the diagonal matrix with diagonal entries  $\epsilon_1, \dots, \epsilon_{n+1}$ , and  $G$  and  $B$  are the matrices computed in the previous two steps. Let  $w$  denote the output.
- (5) Compute in parallel the vector  $v := W^\top w$ . Let  $(v_{00}, v_{01}, \dots, v_{0,d_0-1}, \dots, v_{\ell,0}, v_{\ell,1}, \dots, v_{\ell,d_\ell-1})$  denote its components.
- (6) Output  $F(x, y) = \sum_{i=0}^\ell F_i(x)y^i$  where  $F_i(x) = v_{i,0} + v_{i,1}x + \dots + v_{i,d_i-1}x^{d_i-1}$ .

**THEOREM 7.6.** *The above algorithm correctly computes its output with  $O(\ell n \log_q(n)^2)$  operations on a PRAM with  $n + 1$  processors.*

**PROOF.** The matrix  $VW^\top$  satisfies the displacement equation 26 with the matrices  $G$  and  $B$  computed in steps (2) and (3). Step (4) then produces a nontrivial element in the kernel of  $VW^\top$ . This shows that the vector  $v$  computed in Step (5) is indeed a nontrivial element in the right kernel of  $V$ , and proves correctness of the algorithm.

The finite field  $\mathbb{F}_{q^d}$  can be constructed probabilistically in time proportional to  $d^3 \log(qd) \log(d)$  on one processor [12, Th. 14.42]. Arithmetic operations in  $\mathbb{F}_{q^d}$  can be simulated using  $d^2$  operations over  $\mathbb{F}_q$  on each processor. In the following we thus assume that the processors on the PRAM are capable of executing one arithmetic operation in  $\mathbb{F}_{q^d}$  in  $O(d^2)$  steps. In Step (2)  $G_1$  can be calculated with  $O(\alpha \log(n))$   $\mathbb{F}_{q^d}$  operations on  $n$  processors (each of which computes the rows of  $G_1$  independently). Likewise,  $v_1$  can be computed with  $O(\log(n))$  operations on  $n$  processors. Hence, computation of  $G$  requires  $O(\ell \log(n))$  operations over  $\mathbb{F}_{q^d}$ . Given the special structure of the matrix  $B_1$ ,  $\tilde{B}$  can be computed with  $O(n)$   $\mathbb{F}_{q^d}$  operations on  $n + 1$  processors.  $G_2$  can be computed with  $O(\log(n))$  operations on  $n + 1$  processors. Hence, steps (2) and (3) require  $O(\ell \log(n) + n)$  operations on  $n + 1$  processors. Step (4) needs  $O(\ell n)$  operations on  $(n + 1)$  processors by Theorem 7.4. and Step (5) needs  $O(n)$  operations on  $n + 1$  processors. Hence, the total number of  $\mathbb{F}_{q^d}$ -operations of the algorithm equals  $O(\ell n)$  and it can be performed on  $n + 1$  processors.  $\square$

## References

- [1] S. Ar, R. Lipton, R. Rubinfeld, and M. Sudan. Reconstructing algebraic functions from mixed data. In *Proceedings of the 33rd IEEE Symposium on Foundations of Computer Science*, pages 503–512, 1992.
- [2] D. Augot and L. Pecquet. An alternative to factorisation: a speedup for Sudan’s algorithm and its generalization to algebraic-geometric codes. Preprint, 1998.
- [3] E.R. Berlekamp. *Algebraic Coding Theory*. McGraw-Hill, New York, 1968.
- [4] E. R. Berlekamp and L. Welch. Error correction of algebraic block codes. US Patent Number 4,633,470, 1986.
- [5] E.R. Berlekamp. Bounded distance + 1 soft decision Reed-Solomon decoding. *IEEE Trans. Inform. Theory*, 42:704–720, 1996.
- [6] R. Bitmead and B. Anderson. Asymptotically fast solution of Toeplitz and related systems of linear equations. *Linear Algebra and its Applications*, 34:103–116, 1980.
- [7] J.M.Delosme. Algorithms and Implementations for Liner Least Squares Estimation. Ph.D.Thesis, Stanford University, 1982
- [8] H.Dym. Structured matrices, reproducing kernels and interpolation, *Structured Matrices in Mathematics, Computer Science, and Engineering*, AMS series CONTEMPORARY MATHEMATICS, vol. **280**, pp. 3-30, AMS Publications, 2001.
- [9] B.Friedlander, M.Morf, T.Kailath and L.Ljung, New inversion formulas for matrices classified in terms of their distance from Toeplitz matrices, *Linear Algebra and its Applications*, **27** (1979) 31-60.
- [10] G. Golub and C. van Loan Matrix computations, third edition, The Johns Hopkins University Press Ltd., London, 1996.  
Computing roots of polynomials over function fields of curves. In David Joyner, editor, *Coding Theory and Cryptography: from Enigma and Geheimschreiber to Quantum Theory*, pages 214–228. Springer Verlag, 1999.
- [11] S. Gao and M.A. Shokrollahi. Computing roots of polynomials over function fields of curves. In David Joyner, editor, *Coding Theory and Cryptography: from Enigma and Geheimschreiber to Quantum Theory*, pages 214–228. Springer Verlag, 1999.
- [12] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge Univ. Press, Cambridge, 1999.
- [13] I. Gohberg and V. Olshevsky. Fast algorithm for matrix Nehari problem, roceedings of MTNS-93, Systems and Networks: Mathematical Theory and Applications, v.2, Invited and Contributed Papers,edited by U. Helmke, R. Mennicken and J. Sauers, Academy Verlag,1994, p. 687-690.
- [14] I. Gohberg and V. Olshevsky. Fast state-space algorithms for matrix Nehari and Nehari-Takagi interpolation problems. *Integral Equations and Operator Theory*, 20:44–83, 1994.
- [15] V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and algebraic-geometric codes. In *Proceedings of the 39th IEEE Symposium on Foundations of Computer Science*, pages 28–37, 1998.
- [16] G. Heinig Ein Prinzip zur Invertierung einiger Klassen linearer Operatoren, *Proceedings of the 7th Conference on Methods in Mathematical Physics (7th TMP)*, TH Karl-Marx-Stadt 1979, vol.2, pp.45–50.
- [17] G. Heinig and K. Rost. *Algebraic Methods for Toeplitz-like matrices and operators*, volume 13 of *Operator Theory*. Birkhäuser, Boston, 1984.
- [18] T. Høholdt and R. Refslund Nielsen. Decoding Hermitian codes with Sudan’s algorithm. Preprint, Denmark Technical University, 1999.
- [19] T. Høholdt and R. Pellikaan. On the decoding of algebraic-geometric codes. *IEEE Trans. Inform. Theory*, 41:1589–1614, 1995.
- [20] J. Justesen, K.J. Larsen, H.E. Jensen, and T. Høholdt. Fast decoding of codes from algebraic plane curves. *IEEE Trans. Inform. Theory*, 38:111–119, 1992.
- [21] T. Kailath, Some new algorithms for recursive estimation in constant linear systems, *IEEE transactions on Information Theory*, IT-19, Nov. 1973, 750–760.
- [22] T. Kailath and J. Chun, Generalized Displacement Structure for Block-Toeplitz, Toeplitz-Block, and Toeplitz-Derived Matrices, *SIAM J. Matrix Anal. Appl.*, vol. 15, no. 1, pp. 114–128, January 1994.

- [23] T.Kailath, S.Kung and M.Morf, Displacement ranks of matrices and linear equations, *J. Math. Anal. and Appl.*, **68** (1979) 395-407.
- [24] T. Kailath and V. Olshevsky. Fast Gaussian Elimination with Partial Pivoting for Matrices with Displacement Structure *Mathematics of Computation*, 64 No. 212 (1995), 1557-1576.
- [25] T. Kailath and V. Olshevsky. Displacement structure approach to discrete trigonometric transform based preconditioners of G. Strang and T. Chan types. *Calcolo*, 33:191–208., 1996.
- [26] T. Kailath and V. Olshevsky. Unitary Hessenberg matrices and the generalized Parker-Forney-Traub and Björck-Pereyra algorithms for Szegő-Vandermonde matrices. To appear. Can be obtained from <http://www.cs.gsu.edu/~matvro>, 1997.
- [27] T. Kailath and A.H. Sayed. Displacement structure: Theory and applications. *SIAM Review*, 37:297–386, 1995.
- [28] R. Köetter. Fast generalized minimum-distance decoding of algebraic-geometry and Reed-Solomon codes. *IEEE Trans. Inform. Theory*, 42:721–737, 1996.
- [29] F.T. Leighton. *Introduction to Parallel Algorithms and Architectures : Arrays, Trees, Hypercubes*. Academic Press/Morgan Kaufmann, 1992.
- [30] H. Lev Ari. Nonstationary Lattice-Filter Modeling. Ph.D.Thesis, Stanford University, 1983
- [31] H. Lev-Ari, Displacement Structure: Two Related Perspectives, in Communications, Computing, Control and Signal Processing: A Tribute to Thomas Kailath, A. Paulraj, V. Roychowdhury and C. Schaper, eds., pp. 233-241, Kluwer Academic Publishers, Norwell, MA, 1997.
- [32] F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error-Correcting Codes*. North-Holland, 1988.
- [33] M. Morf. *Fast algorithms for multivariable systems*. PhD thesis, Stanford University, 1974.
- [34] M. Morf. Doubling algorithms for Toeplitz and related equations. In *Proceedings of IEEE Conference on Acoustics, Speech, and Signal Processing, Denver*, pages 954–959, 1980.
- [35] V. Olshevsky (editor). Structured Matrices in Mathematics, Computer Science, and Engineering, AMS series CONTEMPORARY MATHEMATICS, vols. **280**, **281**, AMS Publications, 2001.
- [36] V. Olshevsky Pivoting for structured matrices with applications. <http://www.cs.gsu.edu/~matvro>, 1997.
- [37] V. Olshevsky and V. Pan. A superfast state-space algorithm for tangential Nevanlinna-Pick interpolation problem. In *Proceedings of the 39th IEEE Symposium on Foundations of Computer Science*, pages 192–201, 1998.
- [38] V. Olshevsky and M.A. Shokrollahi. A displacement structure approach to list decoding of algebraic-geometric codes. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, pages 235–244, 1999.
- [39] R. Roth and G. Ruckenstein. Efficient decoding of reed-solomon codes beyond half the minimum distance. In *Prceedings of 1998 International Symposium on Information Theory*, pages 56–56, 1998.
- [40] L.A. Sakhnovich, On similarity of operators (in Russian), *Sibirskij Mat. J.* 8, 4 (1972), 8686–883.
- [41] L.A. Sakhnovich, On the integral equation with kernel depending on the difference of the arguments (in Russian), *Matem. Issledovanya*, Kishinev, 8, 2 (1973), 138–146.
- [42] L.A. Sakhnovich, The factorization of the operator-valued transfer functions, *Soviet Math. Dokl.*, 17 (1976), 203–207.
- [43] L.A. Sakhnovich, Factorization problems and operator identities, *Russian Mathematical Surveys*, 41, 1 (1986), 1 – 64.
- [44] M.A. Shokrollahi and H. Wasserman. List decoding of algebraic-geometric codes. *IEEE Trans. Inform. Theory*, 45:432–437, 1999.
- [45] M. Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *J. Compl.*, 13:180–193, 1997.
- [46] L.R. Welch and E.R. Berlekamp. Error correction for algebraic block codes. U.S. Patent 4,633,470, issued Dec. 30, 1986.

## Appendix A

In this appendix we shall clarify how to implement Step 3 of Algorithm 6.1, i.e., how to find a vector in the right kernel of a matrix  $R := VW^\top$  satisfying

$$JR - R\Delta = GB$$

(cf. (21)). By Lemma 2.4 we could use the extended displacement equation

$$(28) \quad \begin{pmatrix} J & 0 \\ 0 & \Delta \end{pmatrix} \begin{pmatrix} R \\ I \end{pmatrix} - \begin{pmatrix} R \\ I \end{pmatrix} \Delta = \begin{pmatrix} G \\ 0 \end{pmatrix} B$$

to try to run the algorithm 2.5 to compute a vector in the right kernel of  $R := VW^\top$ . Indeed, the latter algorithm is based on Lemma 2.1 which is directly applicable in our situation because  $\begin{pmatrix} J & 0 \\ 0 & \Delta \end{pmatrix}$  is lower triangular, and  $\Delta$  is upper triangular (even diagonal).

However, there are several differences between (28) and (5) that do not allow one to apply the Algorithm 2.5 directly.

- (1) A closer look at (28) reveals that the mapping  $\nabla : \mathbb{F}^{(m+s) \times s} \longrightarrow \mathbb{F}^{(m+s) \times \alpha} \times \mathbb{F}^{\alpha \times s}$  defined by  $\nabla S = \begin{pmatrix} J & 0 \\ 0 & \Delta \end{pmatrix} S - S\Delta$  is not an isomorphism, so the the generator  $\{\begin{pmatrix} G \\ 0 \end{pmatrix}, B\}$  no longer contains the full information about  $\begin{pmatrix} R \\ I \end{pmatrix}$  in (28). Thus, we need to introduce a new definition for a generator,

$$(29) \quad \{\begin{pmatrix} G \\ 0 \end{pmatrix}, B, D\}$$

including in it one more matrix  $D$  to completely describe our structured matrix  $\begin{pmatrix} R \\ I \end{pmatrix}$ .

- (2) Lemma 2.4 shows how to compute a vector in the kernel of  $R$  via computing the Schur complements of  $\begin{pmatrix} R \\ I \end{pmatrix}$ . Further, Lemma 2.1 allows us to speed-up the procedure by manipulation on two generator matrices  $\{\begin{pmatrix} G \\ 0 \end{pmatrix}, B\}$ . In our situation (having a new extended generator (29)) we need to specify a recursion for the third matrix in (29) as well.
- (3) One of the ingredients of the algorithm 2.5 is recovering the first row and the first column of a structured matrix from its generator. We need to clarify this procedure for our new extended generator in (29).
- (4) In what follows we provide formulas resolving the above three difficulties, and specify a modified version of the fast algorithm 2.5 for our matrix  $S := \begin{pmatrix} R \\ I \end{pmatrix}$ . However, the original algorithm 2.5 employs partial row pivoting to run to the completion.<sup>1</sup> Row pivoting was possible due to the

---

<sup>1</sup>Specifically, at each step the algorithm 2.5 uses row interchanges to bring a nonzero element in the (1,1) position of  $S$  (moreover, the absence of appropriate non-zero elements was the stopping criterion).

diagonal form of the matrix  $D$  in (5). In contrast, the new matrix  $J$  in (28) is no longer diagonal, so we have to find a different pivoting strategy.

We next resolve the above four difficulties. Let  $S$  satisfy

$$(30) \quad AS - S\Delta = GB.$$

where we set  $A := \begin{pmatrix} J & 0 \\ 0 & \Delta \end{pmatrix} \in \mathbb{F}^{(n+s) \times (n+s)}$  and  $\Delta \in \mathbb{F}^{s \times s}$  is diagonal.

- (1) A general theory of displacement equations with kernels can be found in [36, 26], and this more delicate case appears in studying *boundary* rational interpolation problems and in the design of preconditioners [25]. As we shall see below, we can recover from  $\{G, B\}$  on the right-hand side of (30) all the entries of the  $(n+s) \times s$  matrix  $S$  but the  $s$  diagonal elements  $\{S(n+1, 1), S(n+2, 2), \dots, S(n+s, s)\}$ . Therefore the triple

$$(31) \quad \{G, B, D\} \quad \text{with} \quad D := \{S(n+1, 1), S(n+2, 2), \dots, S(n+s, s)\}$$

contains the full information on  $S$ , so we call it a *generator*.

It should be mentioned that the idea of generator is in compressing the information on  $n^2$  entries of  $S$  into just  $O(n)$  entries of its generator. The newly defined generator in (31) involves just  $s$  more parameters (i.e., the entries of the  $1 \times s$  row  $D$ ) and therefore is still a very efficient representation.

- (2) Each step of the algorithm 2.5 consists of two parts: (1) recovering the first row and column of  $S$  from its generator; (2) computing a generator  $\{G_2, B_2, D_2\}$  for the Schur complement  $S_2$  of  $S$ . We shall discuss part (1) in the next item, and here we address part (2). A closer look at lemma 2.1 reveals that it applies in our situation and the formulas for computing  $\{G_2, B_2\}$  remain valid. It remains to show how to compute the row  $D_2 = \{d_1^{(2)}, d_2^{(2)}, \dots, d_s^{(2)}\}$  capturing the  $\{(n+1, 1), (n+2, 2), \dots, (n+s-1, s-1)\}$  elements of  $S_2$ . Since  $S_2 = S_{22} - S_{21}S_{11}^{-1}S_{12}$  is the Schur complement of  $S = \begin{pmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{pmatrix}$ , we have the following formulas

$$(32) \quad d_k^{(2)} = S_{n+k, k} - S_{n+k, 1}S_{1, 1}^{-1}S_{1, k},$$

where all the involved entries  $S_{ij}$  of  $S$  are available:  $\{S_{n+k, 1}, S_{1, 1}, S_{1, k}\}$  appear either in the first row or in the first column (we shall show next how to recover them from (31)), and  $S_{n+k, k}$  is an entry of the matrix  $D$  in (31).

- (3) Now we are able to provide formulas to recover the first row and column of  $S$  in (30) from its generator in (31).

The elements of the top row can be obtained by the formula

$$(33) \quad S(1, k) = \frac{G(1, :)B(:, k)}{A(1, 1) - \Delta(1, k)},$$

where we follow the MATLAB notation and denote by  $G(1, :)$  the first row of  $G$  and by  $B(:, k)$  the  $k$ -th column of  $B$ . This formula works only if  $A_{1,1} \neq \Delta_{1,k}$ . If  $A_{1,1} = \Delta_{1,k}$  (this may happen only for  $k = 1$ ) then  $S_{1,1}$  cannot be recovered from  $\{G, B\}$  and is therefore stored in the third matrix  $D$  of (31).

Since  $A$  is a bi-diagonal matrix, the elements of the first column of  $S$  can be recovered as follows:

$$(34) \quad S(1, 1) = \frac{G(1, :)B(:, 1)}{A(1, 1) - \Delta(1, 1)}, \quad S(k, 1) = \frac{[G(k, :) - A(k, k-1)G(k-1, :)]B(:, 1)}{A(k, k) - \Delta(1, 1)}$$

- (4) The above items describe several modifications to the algorithm 2.5. The modified algorithm terminates successfully after  $k$  steps when the first  $m - k$  entries of the  $(m - k + s) \times s$  Schur complement are zero (cf. Lemma 2.4)). However, we need to implement an appropriate pivoting strategy to ensure that at each step of the algorithm, the element  $S(1, 1)$  of the Schur complement is non-zero. Pivoting is the standard method to bring a non-zero element in the  $(1, 1)$  position, however, arbitrary pivoting can destroy the structure of  $S$  and thus block further efficient steps of the algorithm. Indeed, lemma 2.1 requires that the matrix  $A$  is lower triangular and the matrix  $\Delta$  in (30) is upper triangular, which can be destroyed by row/column interchanges. In our case the matrix  $\Delta$  is diagonal, so any  $(P^\top \Delta P)$  is diagonal as well. Therefore the permuted  $(SP)$  satisfies

$$A(SP) - (SP)(P^\top \Delta P) = G(BP).$$

and thus pivoting can be easily implemented in terms of operations on the generator of  $S$ . However, the above column pivoting does not remove all difficulties, and it may happen that the entire first row of  $S$  is zero although there are still non-zero entries in the first column of  $S$ . In this case one may proceed as follows. Since the top row of  $S$  is zero, we are free to make any changes in the first column of  $A$  in (30). Let us zero all the entries of  $A(:, 1)$  but  $A(1, 1)$ . Then  $A$  becomes a block-diagonal matrix. Denote by  $Q$  a permutation that cyclically shifts rows 1 through  $s$  of  $A$ . Then (30) implies

$$(QAQ^\top)(QS) - (QS)\Delta = (QG)B.$$

and  $(QAQ^\top)$  is still lower triangular.

Thus applying column pivoting we bring the non-zero entries in the first row to the  $(1, 1)$  position, and applying row pivoting we move all zeros to the bottom. Note that we do not need to re-compute the entries  $(n+1, 1), \dots, (n+s-1, s-1)$  of  $QS$ , since they are the same as the corresponding entries of  $S$ . Further, since the zero rows remain zero in the Schur complement of  $S$ , they are not considered later in the process of the algorithm, this modified version has a running time of  $O(\alpha n(n+s))$ , where  $\alpha$  is the displacement rank of  $S$ .

Altogether, we obtain the following algorithm:

**ALGORITHM A.1.** *On input integers  $n, s$ ,  $n < s$ , a diagonal matrix  $\Delta \in \mathbb{F}_q^{s \times s}$  with diagonal entries  $x_1, \dots, x_s$ , a lower triangular Jordan matrix  $J \in \mathbb{F}_q^{n \times n}$ , a matrix  $G \in \mathbb{F}_q^{(n+s) \times \alpha}$ , a matrix  $B \in \mathbb{F}_q^{\alpha \times s}$ , and a vector  $d := (d_1, \dots, d_s)$ , the algorithm computes a nonzero vector  $v \in \mathbb{F}_q^s$  such that the matrix  $S$  given by the displacement equation*

$$AS - S\Delta = GB, \quad A := \begin{pmatrix} J & 0 \\ 0 & \Delta \end{pmatrix},$$

and having entries  $(n+1, 1), \dots, (n+s, s)$  equal to  $d_1, \dots, d_s$ .

- (1) Set  $G_1 := G$ ,  $B_1 := B$ ,  $d^1 := d$ ,  $\Delta_1 := \Delta$ , and  $A_1 := A$ ,  $S_1 := S$ .
- (2) For  $i = 1, \dots, n$  do
  - (a) Compute the first row  $(r_1, \dots, r_{s-i+1})$  and the first column  $(c_1, \dots, c_{n+s-i+1})^\top$  of  $S_i$  using (33) and (34).
  - (b) If  $c_1 = \dots = c_{n-i+1} = 0$ , then output  $(c_{n-i+2}, \dots, c_{n+s-i+1})^\top$  and STOP.
  - (c) If the first row is nonzero, find smallest  $k$  such that  $r_k \neq 0$ , interchange  $r_1$  and  $r_k$ , columns 1 and  $k$  of  $B_i$ , and diagonal entries 1 and  $k$  of  $\Delta_i$ .
  - (d) While the first row of  $S_i$  is zero, cyclically shift  $c_1, \dots, c_{n-i+1}$ , rows  $1, \dots, n-i+1$  of  $G_i$ , delete position  $(2, 1)$  of  $A_i$  and interchange diagonal entries 1 and  $n-i+1$  of  $A_i$ . At each step, compute the first row of the new  $S_i$ . (This while loop stops since the first column of  $S_i$  is nonzero.)
  - (e) For  $j = 2, \dots, n+s-i+1$  replace row  $j$  of  $G_i$  by  $-c_j/c_1$  times the first row plus the  $j$ -th row. Set  $G_{i+1}$  as the matrix formed from  $G_i$  by deleting the first row.
  - (f) For  $j = 2, \dots, s-i+1$  replace the  $j$ -th column of  $B_i$  by  $-r_j/c_1$  times the first column plus the  $j$ -th column. Set  $B_{i+1}$  as the matrix formed from  $B_i$  by deleting the first column.
  - (g) Compute the new values  $d^{i+1} = (d'_1, \dots, d'_s)$  from (32).
  - (h) Set  $\Delta_{i+1}$  as the diagonal matrix with entries  $x_{i+1}, \dots, x_s$ . Set  $A_{i+1}$  as the matrix obtained from  $A$  by deleting the first row and the first column. Let  $S_{i+1}$  denote the matrix satisfying the displacement equation  $A_{i+1}S_{i+1} - S_{i+1}\Delta_{i+1} = G_{i+1}B_{i+1}$  and having entries  $(n-i+2, 1), \dots, (n+s-i+1, s)$  equal to  $d'_1, \dots, d'_s$ .

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF CONNECTICUT, STORRS, CT 06269  
*E-mail address:* olshevsky@math.uconn.edu

DIGITAL FOUNTAIN, 39141 CIVIC CENTER DRIVE, FREMONT, CA 94538  
*E-mail address:* amin@digitalfountain.com

# Some Convergence Estimates For Algebraic Multilevel Preconditioners

Matthias Bollhöfer and Volker Mehrmann

**ABSTRACT.** We discuss the construction of algebraic multilevel preconditioners for the conjugate gradient method and derive explicit and sharp bounds for the convergence rates. We present several numerical examples that demonstrate the efficiency of the preconditioner.

## 1. Introduction

For the solution of large sparse linear systems of the form

$$(1) \quad Ax = b, \quad A \in \mathrm{GL}(n, \mathbb{R}), \quad b \in \mathbb{R}^n,$$

sparse approximate inverses [17, 16, 7, 12, 3, 18] have been successfully employed as preconditioners for Krylov–subspace methods [9, 19, 11]. The general principle in these methods is to construct a matrix  $B$  that is sparse and approximates  $A^{-1}$ . Several techniques have been developed, such as minimizing the norm of  $\|AB - I\|$  subject to some prescribed pattern [16, 7, 12] or biconjugate techniques that are based on approximate factorizations  $Z^\top AW \approx D$ , where  $Z, W$  are upper triangular matrices and  $D$  is diagonal [3, 4].

These techniques work well for large classes of matrices, but there are cases when the sparse approximate inverse needs a large number of nonzero entries to become a suitable approximation to the inverse of  $A$ . Also, in the case of approximate inverses based on norm–minimizing techniques, it happens frequently that many singular values of the residual matrix  $E = I - AB$  are quite small, while a small number of singular values are big and stay big even when more fill-in for  $B$  is allowed.

Consider the following example.

**EXAMPLE 1.** For the symmetric positive definite matrix LANPRO/NOS2 from the Harwell–Boeing collection [8], we apply the sparse approximate inverse suggested in [17, 16] with the sparsity pattern of  $A^k$ ,  $k = 0, 1, 2, 3$ . This approach constructs an approximate inverse  $L$  of the Cholesky factor  $\tilde{L}$  of  $A^{-1}$ , i.e.,  $\tilde{L} \approx L$  where  $\tilde{L}\tilde{L}^\top = A^{-1}$ . To evaluate the quality of the preconditioner, we set  $E = I - \omega L^\top AL$

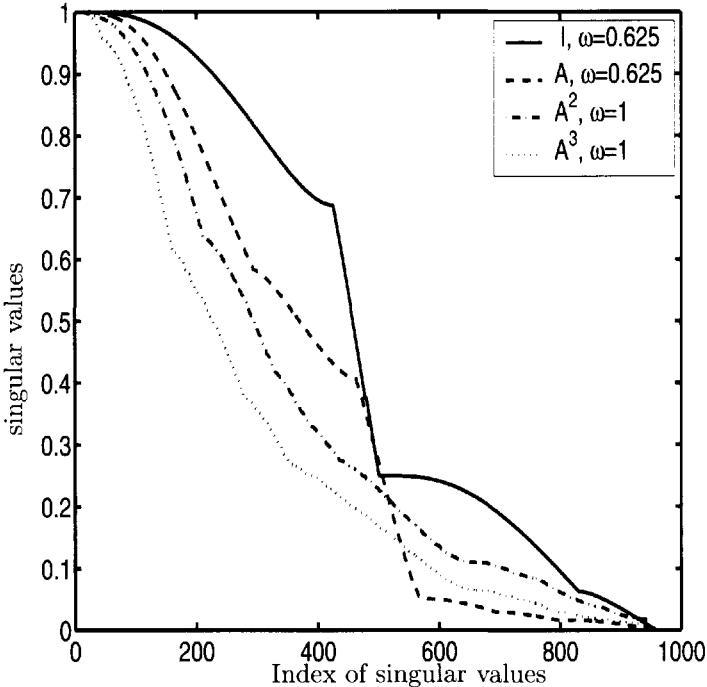
---

1991 *Mathematics Subject Classification.* Primary 65F05; Secondary 65F10, 65F50, 65Y05.

*Key words and phrases.* sparse approximate inverse, sparse matrix, algebraic multilevel method, preconditioning, conjugate gradient method, Krylov subspace method.

for a suitably chosen parameter  $\omega$ , i.e.,  $E$  is our residual, adapted to the symmetric positive case. The scaling parameter  $\omega$  is chosen, so that as many eigenvalues as possible of  $L^T A L$  are close to 1. In Figure 1 we display the singular values of  $E$  in decreasing order. We observe that most of the singular values tend to 0, while a few singular values stay close to 1 even when increasing the number of nonzeros for  $L$ .

FIGURE 1. Singular values of the residual matrix  $E$



The observation that many singular values are small but some stay large, even when the fill-in in the approximate inverse is increased, can be interpreted as follows. The matrix  $B$  approximates  $A$  well on a subspace of large size, while there is almost no approximation on the complementary subspace. In the numerical treatment of partial differential equations this effect is typically called *smoothing property* [13].

If, after scaling the norm to 1, many of the singular values of the matrix  $E = I - AB$  are small, while the others are close to 1, then it follows that

$$(2) \quad E = I - AB = UV^\top + F = E_0 + F,$$

with matrices  $U, V^\top \in \mathbb{R}^{n,r}$  and  $\|F\| \leq \eta < 1$ , i.e., the residual  $E = I - AB$  can be approximated well by a matrix of rank  $r$  much smaller than  $n$ . Here realistically  $r \leq cn$  with  $c \approx 0.5$ . We want to use such an approximative factorization  $UV^\top$  for the preconditioning of large scale problems. If the rank  $r$  of the approximation is of this order, then, to keep the storage requirements small, it would be ideal to have a sparse representation of the approximation  $UV^\top$ . This does not necessarily mean that  $U, V$  have to be sparse, e.g. we could have  $E_0 = PZ^{-1}\hat{P}^\top$  with sparse matrices  $P, \hat{P}, Z$ , while their product need not be sparse at all.

In order to determine such a sparse factorization, we observe that if  $\eta \ll 1$ , then the entries of  $E_0$  only slightly differ from those of  $E$ . So we may expect that an appropriate selection of columns of  $E$  will be a good choice for determining  $UV^\top$ . This is underscored by the following lemma.

**LEMMA 1.** [6]. *Let  $E \in \mathbb{R}^{n,n}$  and let  $E = [U_1, U_2] \text{ diag } (\Sigma_1, \Sigma_2) [V_1, V_2]^\top$  be the singular value decomposition of  $E$  with  $U_1, V_1$  having  $r$  columns and  $\|\Sigma_2\|_2 \leq \varepsilon$ . Then there exist a permutation matrix  $\Pi = [\Pi_1, \Pi_2]$  with analogous partitioning such that*

$$(3) \quad \inf_{Z \in \mathbb{R}^{r,r}} \|U_1 Z - E \Pi_1\| \leq \varepsilon.$$

By applying Lemma 1 to  $E^\top$  instead of  $E$  we can analogously approximate  $V_1$  by suitably chosen rows of  $E$ .

In [6] it has been discussed how one can use approximate factorizations such as (2) to modify a given preconditioner  $B$ . Instead of iterating with the preconditioned matrix  $AB$  one uses the updated preconditioned matrix  $AB(I + (AB)^{-1}UV^\top)$ . In order to avoid that a small error, between the exact subspace  $U$  (associated with large singular values of  $E$ ) and a set of columns of  $E$ , blows up the norm of  $UV^\top$ , one determines a decomposition of the form

$$(4) \quad E = (AB)PS + F.$$

Here we require that  $\|F\| \leq \eta$  and  $P, S^\top \in \mathbb{R}^{n,r}$  to keep the error  $AB(I + (AB)^{-1}UV^\top) - I = AB(I + PS) - I$  small. But since  $E = I - AB$ , this means that we have to determine a factorization

$$(5) \quad AB(I + PS) = I - F$$

with  $\|F\| \leq \eta$ .

For the analysis of such a preconditioner, we need to study the approximation properties of the correction term  $I + PS$  in (5). A detailed analysis and sharp convergence bounds that extend results of [6] are given in Section 3.

Furthermore, to obtain an efficient linear system solver, in (4) we have to find a representation  $PS = P(Z^{-1}\hat{P}^\top)$  with sparse matrices  $P, \hat{P}, Z$  and we also have to find an efficient method to determine the desired columns of  $E$  as suggested by Lemma 1. Both topics have been discussed in [6]. To determine the columns of  $E$  for the construction of the preconditioner, one could use a  $QR$ -like decomposition with column pivoting of  $E\Pi = QR$  and use as  $P$  the first  $r$  columns of  $Q$ . A more elegant choice would be to use instead of the first  $r$  columns of  $Q$  the first  $r$  columns of  $E\Pi$ , since their columns span the same space as the associated columns of  $Q$ . But the computation of such a  $QR$ -like decomposition, i.e., in particular the computation of the pivoting matrix  $\Pi$ , has to be carried out with small memory requirements.

Once a strategy for constructing and updating sparse approximate inverse preconditioners has been established, this approach can be applied in a recursive way, leading to algebraic multigrid type methods. These methods are discussed in Section 2. Essential for the success of these multilevel processes is the coarsening process that we discuss in Section 4. We illustrate our results with several numerical examples.

In the sequel for symmetric matrices  $A, B$  we will use the notation  $A \geq B$ , if  $A - B$  has nonnegative eigenvalues. When talking about a matrix  $P$  we do not

strictly distinguish between the matrix  $P$  itself and the space  $\{Px : x \in \mathbb{R}^r\}$  that is spanned by its columns.

## 2. Algebraic Multilevel Preconditioners

In this section we briefly review two types of algebraic multilevel preconditioners for symmetric positive definite systems that were introduced in [6]. Since here we concentrate on the symmetric case, we focus on preconditioners  $B = LL^\top$  and use the symmetrized version of residual matrix  $E = I - L^\top AL$ . The symmetrized version of (4) then is as

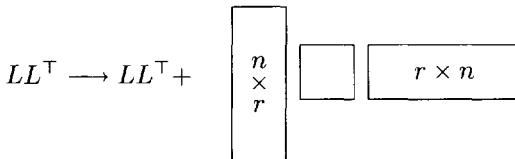
$$E = (L^\top AL)PS + F.$$

Let  $A \in \mathbb{R}^{n,n}$  be symmetric positive definite. Suppose that  $LL^\top$  is a symmetric positive definite matrix that is an approximation to  $A^{-1}$  and that we use as preconditioner for the conjugate gradient method [10]. Here it is understood that  $L$  is a sparse matrix, so that linear systems can be solved with low complexity on the available computer architecture. Suppose that one realizes during the iteration process that the approximation of  $A^{-1}$  by  $LL^\top$  is not satisfactory, i.e., that the condition number of the preconditioned matrix  $M = L^\top AL$  is not small enough to get fast convergence of the conjugate gradient method.

In order to improve the preconditioner one can determine a matrix of the form

$$(6) \quad M^{(1)} = L(I + PZ^{-1}P^\top)L^\top$$

with  $P \in \mathbb{R}^{n,p}$ ,  $Z \in \mathbb{R}^{r,r}$  nonsingular, sparse and  $r$  smaller than  $n$ , so that  $M^{(1)}$  is a better approximation to  $A^{-1}$  than  $LL^\top$ . In (6) the sparse approximate inverse  $LL^\top$  is augmented as



Note that one does not have to assume that  $P$  has low rank, but only that  $r \leq cn$  with  $c < 1$ , e.g.  $c = 0.5$ . It is very important, however, that  $P, Z$  are sparse.

The particular form (6) resembles that of an algebraic two-level method, where multiplication with  $P, P^\top$  corresponds to the mapping between fine and coarse grid and  $Z$  represents the coarse grid system. Note further, that in using  $L(I + PZ^{-1}P^\top)L^\top$  as a preconditioner for  $A$ , only a system with  $Z$  has to be solved.

As shown in Lemma 1, skilfully chosen columns/rows of the residual matrix  $E = I - M = I - L^\top AL$  can be used to approximate the invariant subspace of  $E$  associated with its large eigenvalues. As will be shown below, precisely this invariant subspace has to be approximated by  $P$ . In the sense of the underlying undirected graph of  $E$ , we refer to those nodes as *coarse grid nodes* whose columns/rows of  $E$  will be used to approximate the invariant subspace of  $E$  associated with its largest singular values while the remaining nodes are called *fine grid nodes*. The process of detecting a suitable set of coarse grid nodes will be called *coarsening process*. Once one has selected the coarse grid nodes, they are in a natural way embedded in the initial graph. In addition the subset of coarse grid nodes has its own related graph. It has been shown for special case in [6] that among all symmetric positive preconditioners for  $A$  of type  $M = L(I + PZ^{-1}P^\top)L^\top$  with fixed  $L$  and  $P$  the

choice  $Z = P^\top MP$  is (almost) optimal in the sense of quadratic forms. In this case the graph of  $Z$  describes a natural graph associated with the coarse grid nodes. We will call it *coarse grid* in analogy to the graph associated with a grid in the numerical solution of partial differential equations.

Recalling the well-known techniques of constructing good preconditioners for the conjugate gradient method applied to symmetric positive definite systems, e.g. [10, 14, 20], we should choose  $P$  and  $Z$  such that

$$(7) \quad \mu A \leq \left( M^{(1)} \right)^{-1} \leq \mu \kappa^{(1)} A$$

with  $\kappa^{(1)}$  as small as possible and  $\mu > 0$ . Here  $\kappa^{(1)} \geq 1$  is the condition number of  $M^{(1)}A$ , i.e., the ratio of the largest by the smallest eigenvalue of  $M^{(1)}A$  and thus  $\kappa^{(1)} = 1$  would be optimal. It is well-known that a small condition number is essential for fast convergence of the conjugate gradient method [10].

For a discretized elliptic partial differential equation one can construct optimal preconditioners (with a condition number that does not depend on the fineness of the discretization) using multigrid methods [14]. In order to obtain a similar preconditioner from sparse approximate inverses, consider the use of  $LL^\top$  in a linear iteration scheme with initial guess  $x^{(0)} \in \mathbb{R}^n$ . The iteration scheme [22] for the solution of  $Ax = b$  has the form

$$x^{(k+1)} = x^{(k)} + LL^\top(b - Ax^{(k)}), \quad k = 0, 1, 2, \dots$$

The error propagation matrix  $I - LL^\top A$  satisfies  $x - x^{(k+1)} = (I - LL^\top A)(x - x^{(k)})$ . In multilevel techniques [13] one uses this iteration for pre and post smoothing and in addition one has to add a coarse grid correction. In terms of the error propagation matrix this means that instead of  $I - LL^\top A$  we have  $(I - LL^\top A)(I - \hat{P}Z^{-1}\hat{P}^\top A)(I - LL^\top A)$  as error propagation matrix, where  $\hat{P} = LP$ . This product can be rewritten as  $I - M^{(2)}A$  with

$$(8) \quad \begin{aligned} M^{(2)} &= 2LL^\top - LL^\top ALL^\top + (I - LL^\top A)\hat{P}Z^{-1}\hat{P}^\top(I - ALL^\top) \\ &= L(2I - M + EPZ^{-1}P^\top E)L^\top, \end{aligned}$$

where  $M = L^\top AL$  and  $E = I - M$ . Note that when applying  $M^{(2)}$  to a vector  $x$ , the matrices  $A$  (or  $M$ ) and  $LL^\top$  have to be applied only twice, i.e., the application of this operator is less expensive than it looks. Again one is interested in choosing  $P, Z$  such that

$$(9) \quad \mu A \leq \left( M^{(2)} \right)^{-1} \leq \mu \kappa^{(2)} A$$

with  $\kappa^{(2)}$  as small as possible.

To derive the approximation properties of  $M^{(1)}, M^{(2)}$  we first discuss the optimal factors  $P, Z$  for given  $A, L$ . We use the spectral decomposition of the residual matrix

$$(10) \quad E = I - L^\top AL = I - M = V\Lambda V^\top,$$

where  $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$  with diagonal elements in decreasing order and  $V = [v_1, \dots, v_n]$  is orthogonal.

For a given fixed rank  $r$ , it has been shown in [2, 6] that

$$(11) \quad P := [v_1, \dots, v_r], \quad Z := P^\top MP$$

gives an (almost) optimal approximation, i.e., we obtain the minimal  $\kappa^{(1)}$  in (7). But this observation is more of theoretical interest, since in practice the spectral decomposition is not available and even if it were available, then it would be very expensive to apply, since the matrix  $P$  would be a full matrix. Instead we would like to determine  $P, Z$  that are inexpensive to apply and still produce good approximation properties in  $M^{(1)}$  and  $M^{(2)}$ . The specific choice  $Z = P^\top MP$  then corresponds to an exact two-level method. A more general choice of  $Z$  should satisfy the relation

$$\gamma Z \leq P^\top MP \leq \Gamma Z$$

to fit into the framework which we will present in Section 3. One can read  $Z$  as a perturbed coarse grid system or a perturbed 2-level method, when  $P^\top MP$  is not solved exactly. The common choice to perturb  $P^\top MP$  would be to substitute  $(P^\top MP)^{-1}$  recursively by further multilevel steps. To do this we replace the term  $Z^{-1}$  in

$$(12) \quad L(I + PZ^{-1}P^\top)L^\top, \text{ where } Z = P^\top MP,$$

by a further additive approximation  $L_1 \left( I + P_1 A_{H,1}^{-1} P_1^\top \right) L_1^\top$ , where  $L_1 L_1^\top$  is a factored sparse approximate inverse of the exact coarse grid system  $P^\top MP$ ,  $P_1$  is used to define the next coarsening step and  $A_{H,1} = (P_1^\top L_1^\top Z L_1 P_1)$ . A three-level preconditioner then has the form

$$\begin{aligned} M_2^{(1)} &= L \left( I + P \left( L_1 \left( I + P_1 A_{H,1}^{-1} P_1^\top \right) L_1^\top \right) P^\top \right) L^\top \\ &= LL^\top + LPL_1L_1^\top P^\top L^\top + LPL_1P_1A_{H,1}^{-1}P_1^\top L_1^\top P^\top L. \end{aligned}$$

For the construction of higher levels, the procedure is analogous. This leads to the following algebraic multilevel schemes.

**DEFINITION 1.** Let  $A \in \mathbb{R}^{n,n}$  be symmetric positive definite and let  $n = n_l > n_{l-1} > \dots > n_0 > 0$  be integers. For chosen full rank matrices  $\hat{P}_k \in \mathbb{R}^{n_k \times n_{k-1}}$ ,  $k = l, l-1, \dots, 1$ , define  $A_k$  via

$$A_k = \begin{cases} A & k = l \\ \hat{P}_{k+1}^\top A_{k+1} \hat{P}_{k+1} & k = l-1, l-2, \dots, 0. \end{cases}$$

Choose nonsingular  $L_k \in \mathbb{R}^{n_k \times n_k}$  such that  $L_k L_k^\top \approx A_k^{-1}$ ,  $k = 0, \dots, l$ . Then multilevel sparse approximate preconditioners  $M_l^{(1)}, M_l^{(2)}$  are recursively defined for  $k = 0, 1, 2, \dots, l$  via

$$(13) \quad M_k^{(1)} = \begin{cases} A_0^{-1} & k = 0 \\ L_k L_k^\top + \hat{P}_k M_{k-1}^{(1)} \hat{P}_k^\top & k > 0 \end{cases}$$

and

$$(14) \quad M_k^{(2)} = \begin{cases} A_0^{-1} & k = 0 \\ L_k (2I - L_k^\top A_k L_k) L_k^\top + (I - L_k L_k^\top A_k) \hat{P}_k M_{k-1}^{(2)} \hat{P}_k^\top (I - A_k L_k L_k^\top) & k > 0 \end{cases}$$

For  $l = 1$  we obviously obtain the operators  $M^{(1)}$  and  $M^{(2)}$ . If one exactly decomposes  $A_0^{-1} = L_0 L_0^\top$ , e.g. by Cholesky decomposition and sets  $\Pi_k =$

$\hat{P}_l \hat{P}_{l-1} \cdots \hat{P}_{k+1}$  then one obtains [6]

$$(15) \quad M_l^{(1)} = \sum_{k=0}^l \Pi_k L_k L_k^\top \Pi_k^\top$$

and

$$(16) \quad I - M_l^{(2)} A = (I - \Pi_l L_l L_l^\top \Pi_l^\top A) \cdots (I - \Pi_0 L_0 L_0^\top \Pi_0^\top A) \cdots (I - \Pi_l L_l L_l^\top \Pi_l^\top A).$$

One sees from (15) and (16) that  $M_l^{(1)}$  can be viewed as *additive multilevel method*, since all the projections  $\Pi_k$  are formally performed simultaneously, while  $M_l^{(2)}$  can be viewed as *multiplicative multilevel method*, since the projections  $\Pi_k$  are performed successively.

Operator  $M_l^{(2)}$  is immediately derived from the  $V$ -cycle method in partial differential equations [13]. For operator  $M_l^{(1)}$  a special case occurs when  $L_k L_k^\top = \frac{1}{\alpha_k} I$ . In this case  $E_k = I - \alpha_k A_k$  and choosing columns of  $E_k$  can be expressed as applying a permutation  $\Phi_k \in \mathbb{R}^{n_k, n_{k-1}}$  to  $E_k$ , i.e.  $P_k = (I - \alpha_k A_k)\Phi_k$ . In this case  $M_l^{(1)}$  reduces to

$$M_l^{(1)} = \frac{1}{\alpha_l} (I + \alpha_l P_l M_{l-1} P_l^\top) = \frac{1}{\alpha_l} \left( I + \frac{\alpha_l}{\alpha_{l-1}} P_l (I + \alpha_{l-1} P_{l-1} M_{l-2} P_{l-1}^\top) P_l^\top \right).$$

For this type of operator in [15] optimal choices for  $\alpha_k$  have been discussed with respect to an a priori chosen permutation matrix  $\Phi_k$ . This kind of operator has also been studied in [1, 2].

In summary, the construction of updated preconditioners can be interpreted as a multilevel scheme. In view of this interpretation we need an analysis of the approximation properties of the resulting operators. In the following section we extend the analysis that was introduced in [6] and sharpen the bounds derived there.

### 3. Approximation Properties

In this section we discuss the approximation properties of  $M^{(1)}$  and  $M^{(2)}$  from (6), (8) for the case  $l = 1$  and an approximate coarse grid system  $Z \approx \hat{P}^\top A \hat{P}$ . To simplify notation we will drop the index  $k$  from Definition 1.

We first recall the following result:

**THEOREM 2.** [6] Let  $A \in \mathbb{R}^{n,n}$  be symmetric positive definite and let  $L \in \mathbb{R}^{n,n}$  be nonsingular such that  $M = L^\top A L \leq I$ . Let  $P \in \mathbb{R}^{n,r}$  with  $\text{rank}(P) = r$ ,  $\hat{P} = LP$  and let  $W \in \mathbb{R}^{n,n-r}$  have  $n-r$  be such that  $W^\top MP = 0$ . Let, furthermore,  $Z \in \mathbb{R}^{r,r}$  be symmetric positive definite such that

$$(17) \quad \gamma P^\top MP \leq Z \leq \Gamma P^\top MP$$

with positive constants  $\gamma, \Gamma$ .

i) If

$$(18) \quad W^\top W \leq \Delta W^\top MW,$$

then for  $M^{(1)}$  as in (13) we have

$$(19) \quad \frac{\gamma}{\gamma + 1} A \leq \left( M^{(1)} \right)^{-1} \leq \max\{\Gamma, \Delta\} A.$$

ii) If in (17)  $\gamma \geq 1$  and

$$(20) \quad \begin{bmatrix} 0 & 0 \\ 0 & W^\top MW \end{bmatrix} \leq \Delta [P, W]^\top (M - EME) [P, W],$$

then for  $M^{(2)}$  as in (14) we have

$$(21) \quad A \leq \left( M^{(2)} \right)^{-1} \leq \max\{\Gamma, \Delta\} A,$$

where  $E = I - M$ .

For the operator  $M^{(1)}$  one can also estimate the condition number of  $M^{(1)}A$  in terms of the angle between the invariant subspace associated with the  $r$  smallest eigenvalues and  $P$ , see [2]. Note that in Theorem 2 we have  $\Delta \geq 1$ , since  $M \leq I$ . Thus, if we set  $Z = P^\top MP$  then  $\gamma = \Gamma = 1$  and the bounds for  $M^{(1)}$  are determined by  $\Delta$  only. Via (18) we see that the inequality for  $M$  is only needed on the subspace  $W$  which is the  $M$ -orthogonal complement of  $\text{span } P$ .

In the following result we sharpen the bounds for  $M^{(1)}, M^{(2)}$  in Theorem 2.

**THEOREM 3.** *Under the assumptions of Theorem 2 the following holds.*

i) If  $\Delta, \hat{\Delta}$  are constants satisfying

$$(22) \quad W^\top W \leq \Delta W^\top MW, \quad W^\top MW \leq \hat{\Delta} W^\top M^2 W,$$

then for  $M^{(1)}$  as in (13) we have

$$(23) \quad \frac{\gamma}{\gamma+1} A \leq (M^{(1)})^{-1} \leq \max\{\Gamma, 2 \frac{(\Gamma+1)\Delta\hat{\Delta}}{\Delta+\Gamma\hat{\Delta}}\} A.$$

ii) If in (17)  $\gamma \geq 1$  and  $\hat{\Delta}$  is a constant satisfying

$$(24) \quad W^\top MW \leq \hat{\Delta} W^\top (M - EME) W,$$

then for  $M^{(2)}$  as in (14) we have

$$(25) \quad A \leq (M^{(2)})^{-1} \leq \Gamma \hat{\Delta} A.$$

**PROOF.** For the proof of i) set  $\Delta = \max\{\Gamma, 2 \frac{(\Gamma+1)\Delta\hat{\Delta}}{\Delta+\Gamma\hat{\Delta}}\}$ . It suffices to show that

$$(M^2 + MPZ^{-1}P^\top M)^{-1} \leq \Delta M^{-1}.$$

Multiplying with  $[W, P]^{-1}$  from the left and its transpose from the right and using the fact that  $W^\top MP = 0$ , we obtain

$$\begin{pmatrix} W^\top M^2 W & W^\top M^2 P \\ P^\top M^2 W & P^\top M^2 P + P^\top MPZ^{-1}P^\top MP \end{pmatrix}^{-1} \leq \Delta \begin{pmatrix} W^\top MW & 0 \\ 0 & P^\top MP \end{pmatrix}^{-1}.$$

The diagonal blocks of the left hand side matrix are the inverses  $M_{11}^{-1}, M_{22}^{-1}$  of the Schur-complements  $M_{11}, M_{22}$ , where

$$M_{22} = P^\top MPZ^{-1}P^\top MP + P^\top M (I - MW(W^\top M^2 W)^{-1}W^\top M) MP \geq \frac{1}{\Gamma} P^\top MP,$$

and

$$\begin{aligned}
M_{11} &= W^\top M^2 W - W^\top M^2 P (P^\top M^2 P + P^\top M P Z^{-1} P^\top M P)^{-1} P^\top M^2 W \\
&\geq W^\top M^2 W - W^\top M^2 P \left( P^\top M^2 P + \frac{1}{\Gamma} P^\top M P \right)^{-1} P^\top M^2 W \\
&\geq W^\top M^2 W - \frac{\Gamma}{\Gamma+1} W^\top M^2 P (P^\top M^2 P)^{-1} P^\top M^2 W \\
&= \frac{1}{\Gamma+1} W^\top M^2 W + \frac{\Gamma}{\Gamma+1} W^\top M (I - M P (P^\top M^2 P)^{-1} P^\top M) M W \\
&= \frac{1}{\Gamma+1} W^\top M^2 W + \frac{\Gamma}{\Gamma+1} W^\top M (W (W^\top W)^{-1} W^\top) M W \\
&\geq \left( \frac{1}{(\Gamma+1)\hat{\Delta}} + \frac{\Gamma}{(\Gamma+1)\Delta} \right) W^\top M W.
\end{aligned}$$

Since for all symmetric positive definite matrices

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{12}^\top & A_{22} \end{pmatrix} \leq 2 \begin{pmatrix} A_{11} & 0 \\ 0 & A_{22} \end{pmatrix},$$

inequality (23) follows.

For ii), we set  $T = I - M^{1/2} P (P^\top M P)^{-1} P^\top M^{1/2}$ ,  $\tilde{T} = I - M^{1/2} P Z^{-1} P^\top M^{1/2}$  and  $\Delta = \Gamma \hat{\Delta}$ .  $T$  is the exact orthogonal projection to  $M^{1/2} W$ , which is the orthogonal complement of  $M^{1/2} P$  and  $\tilde{T}$  can be interpreted as a special perturbation of  $T$ . We can see easily that

$$I - M^{1/2} L^{-1} M^{(2)} L^{-\top} M^{1/2} = I - (2M - M^2) - E M^{1/2} P Z^{-1} P^\top M^{1/2} E = E \tilde{T} E.$$

In order to show (25), we have to find  $\Delta > 0$  such that

$$\Delta(I - E \tilde{T} E) \geq M^{1/2} L^{-1} A^{-1} L^{-\top} M^{1/2} = I,$$

or equivalently

$$E \tilde{T} E \leq (1 - \frac{1}{\Delta}) I.$$

Note that (24) is equivalent to

$$T E^2 T \leq (1 - \frac{1}{\hat{\Delta}}) I.$$

This can be seen by multiplying with  $M^{1/2}$  on both sides and with  $[P, W]$  from the right and its transpose from the left. This yields

$$T E^2 T \leq (1 - \frac{1}{\hat{\Delta}}) I$$

or equivalently

$$[P, W]^\top M^{1/2} T E^2 T M^{1/2} [P, W] \leq (1 - \frac{1}{\hat{\Delta}}) [P, W]^\top M [P, W].$$

Again this is equivalent to

$$\begin{bmatrix} 0 & 0 \\ 0 & W^\top E M E W \end{bmatrix} \leq (1 - \frac{1}{\hat{\Delta}}) \begin{bmatrix} P^\top M P & 0 \\ 0 & W^\top M W \end{bmatrix}$$

and it follows that

$$ETE = ET^2 E \leq (1 - \frac{1}{\Delta}) I.$$

Since  $\tilde{T} \leq (1 - \frac{1}{\Gamma})I + \frac{1}{\Gamma}T$  it suffices to choose  $\Delta > 0$  such that

$$(1 - \frac{1}{\Gamma})EME + \frac{1}{\Gamma}EM^{1/2}TM^{1/2}E \leq \left(1 - \frac{1}{\Gamma} + \frac{1}{\Gamma}(1 - \frac{1}{\hat{\Delta}})\right)M \leq (1 - \frac{1}{\Delta})M$$

to obtain (25).  $\square$

Note that in Theorem 3, if  $W^\top W \leq \Delta W^\top MW$ , then we already have  $W^\top MW \leq \Delta W^\top M^2W$ . Thus  $\hat{\Delta} \leq \Delta$  and

$$\frac{(\Gamma + 1)\Delta\hat{\Delta}}{\Delta + \Gamma\hat{\Delta}} \leq \Delta.$$

In this sense the bounds of Theorem 3 are sharper than the bounds of Theorem 2. But if  $\hat{\Delta} \ll \Delta$  then we have

$$\frac{(\Gamma + 1)\hat{\Delta}}{1 + \frac{\Gamma\hat{\Delta}}{\Delta}} \leq (\Gamma + 1)\hat{\Delta}$$

and this is almost an equality. So if  $Z$  is scaled such that  $\Gamma = 1$ , then we obtain the sharper bound

$$(M^{(1)})^{-1} \leq 4\hat{\Delta}A.$$

In other words, the inequality  $W^\top MW \leq \hat{\Delta}W^\top M^2W$  gives much better information on the approximation properties than the inequality  $W^\top W \leq \Delta W^\top MW$ .

We have a similar situation for  $M^{(2)}$ . Clearly  $\hat{\Delta} \leq \Delta$ . But  $\hat{\Delta} \ll \Delta$  is possible.

We will demonstrate the difference between the bounds in Theorems 2 and 3 in the following examples.

**EXAMPLE 2.** Consider the problem

$$-(au')' = f \text{ in } [0, 1], \quad u(0) = u(1) = 0,$$

where

$$a(x) = \begin{cases} 10^0 & x \in [0, 1/4) \\ 10^2 & x \in (1/4, 1/2) \\ 10^3 & x \in (1/2, 3/4) \\ 10^4 & x \in (3/4, 1] \end{cases}$$

and on the interfaces  $1/4, 1/2, 3/4$  the mean value is taken. We discretize this problem using piecewise linear finite elements and a uniform mesh size  $h = 1/512$ . We compare the condition number of the preconditioned system matrix for the two-level preconditioners

$$M_i^{(1)} = D(I + P_i A_{H,i}^{-1} P_i^\top)D, \quad M_i^{(2)} = D(2I - M + EP_i A_{H,i}^{-1} P_i^\top E)D,$$

where  $D$  is the positive square root of the diagonal of  $A$  and where  $P_i$  is the matrix of columns  $2, 4, 6, \dots, 2i$  of  $E = I - \frac{1}{2}M$ . Here  $M$  is defined via  $M = DAD$ . We have that  $A_{H,i} = P_i^\top MP_i$  is the coarse grid system. Note that except for the nodes located near  $1/4, 1/2, 3/4$ , the matrix  $E$  essentially corresponds to the matrix  $\text{tridiag}([\frac{1}{4}, \frac{1}{2}, \frac{1}{4}])$ . In Figure 2 we depict the exact condition number of the preconditioned system for preconditioner  $M^{(1)}$  as well as the bounds on the condition numbers obtained by Theorems 2 and 3. The horizontal axis shows the number of columns used in  $P_i$ .

Figure 2 shows that for  $M^{(1)}$  the bound obtained by Theorem 2 is more pessimistic, especially when more columns are used in  $P_i$ . The estimate from Theorem 3 is sharper.

FIGURE 2. Bounds for the condition number when using preconditioner  $M^{(1)}$ . Exact condition number versus  $\Delta$  (Theorem 2) and  $\hat{\Delta}$  (Theorem 3).

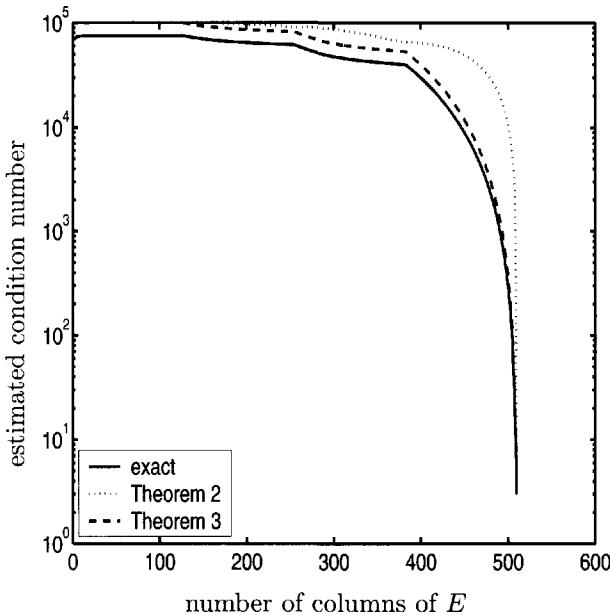


Figure 3 shows the bounds for  $M^{(2)}$ . In this case one can not distinguish between the exact value and both bounds (up to numerical rounding errors), but it should be noted that the bound of Theorem 3 is the easiest one to obtain.

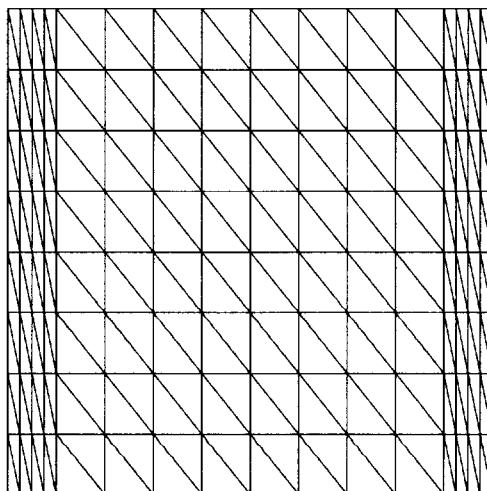
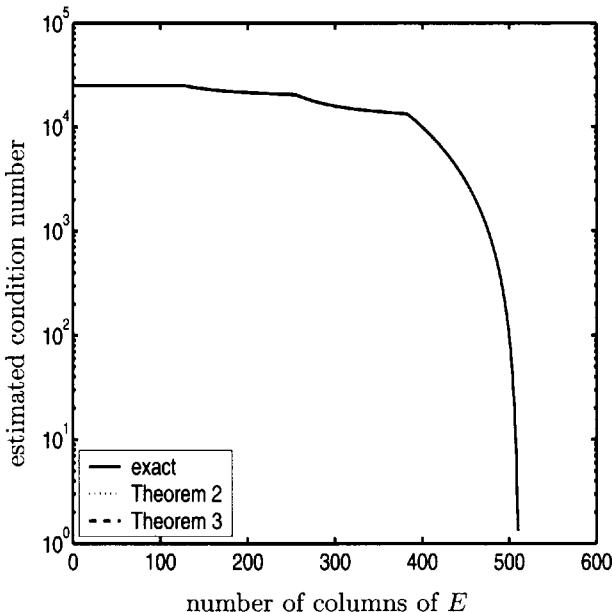
Example 2 illustrates the situation for a simple one-dimensional model problem. Our next example demonstrates that similar results also holds in the two-dimensional case.

EXAMPLE 3. Consider the problem

$$\begin{aligned} -\varepsilon^2 u_{xx} - u_{yy} &= f \text{ in } [0, 1]^2 \\ u &= g \text{ on } \partial[0, 1]^2 \end{aligned}$$

where  $\varepsilon = 10^{-2}$ . Again we use piecewise linear finite elements. The discretization is done using a uniform triangulation with two additional boundary layers of size  $\frac{\varepsilon}{4} \times 1$  near the left and also near the right boundary (see the grid below). Within these boundary layers the triangles are condensed by an additional factor  $\varepsilon/4$  in  $x$ -direction. In  $y$ -direction the mesh size is  $h = 1/32$ .

FIGURE 3. Bounds for the condition number when using preconditioner  $M^{(2)}$ . Exact condition number versus  $\Delta$  (Theorem 2) and  $\hat{\Delta}$  (Theorem 3).

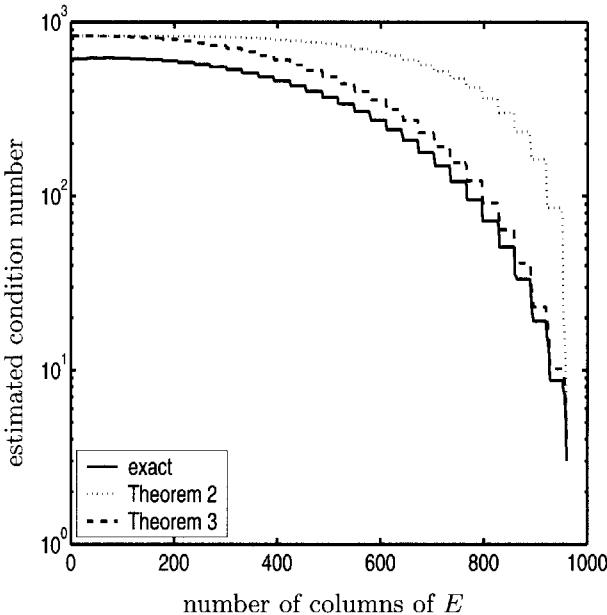


As in Example 2, we compare the exact condition number of the preconditioned system matrix for  $M_i^{(1)}$  and  $M_i^{(2)}$  with the bounds of Theorems 2 and 3.

Figure 4 shows the comparison for the preconditioner  $M^{(1)}$ . It shows again that the bound of Theorem 3 is much better than that of Theorem 2.

In Figure 5 we depict the results for  $M^{(2)}$  where again the bounds and the exact value are almost indistinguishable but again the one from Theorem 3 is the most easy bound to obtain.

FIGURE 4. Bounds for the condition number when using preconditioner  $M^{(1)}$ . Exact condition number versus  $\Delta$  (Theorem 2) and  $\hat{\Delta}$  (Theorem 3)



The bounds obtained by Theorem 3 are in general sharper and easier to obtain than those of Theorem 2, since in Theorem 3 both bounds for  $\hat{\Delta}$  only require the subspace that is given by the  $M$ -orthogonal complement of  $P$ . On the other hand the bounds of Theorem 2 can be more easily generalized to the general case  $l > 1$ , see [6]). It is possible, though more technical, to generalize the sharper bounds of Theorem 3 to the multilevel case ( $l > 1$ ). The difficulties arise from the fact that  $\Gamma$  is not as well isolated in (23) as in (19).

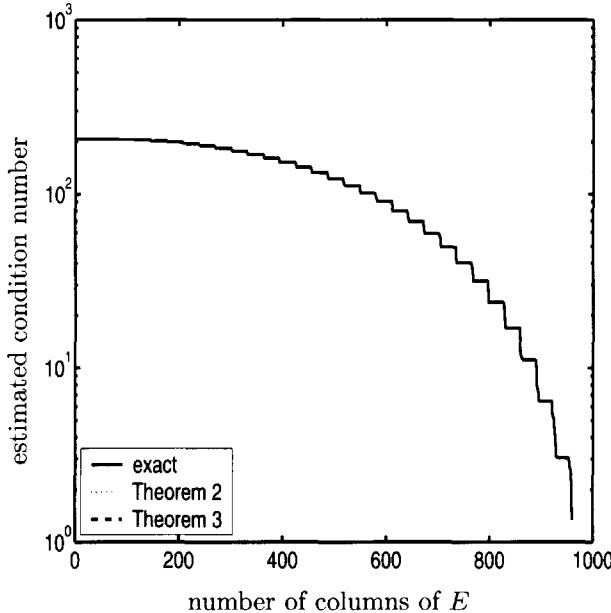
In this section we have derived sharp bounds for the condition numbers of the multilevel preconditioned systems. In practice to determine these bounds still requires the solution of an eigenvalue problem. However, one could estimate  $\hat{\Delta}$ , e.g. by applying a test vector  $x$  to the inequalities (22),(24). For more details see Section 4. The results so far require two spaces (or matrices)  $P$  and  $W$  that are orthogonal with respect to the inner product defined by  $M = L^\top AL$ . One can construct a sequence of well-suited matrices  $P_k$  in each step  $k$  using the results of Theorems 2 and 3. This construction leads to a multilevel hierarchy. This will be the topic of the next section.

#### 4. A Simplified Coarsening Scheme

The approximation bounds that we have derived in Section 3 can be utilized to construct an algebraic multilevel preconditioner by skilfully choosing columns of the residual matrix  $E$ . Here we focus only on a sketch of the main idea and the basic ingredients.

The basic components of the construction are:

FIGURE 5. Bounds for the condition number when using preconditioner  $M^{(2)}$ . Exact condition number versus  $\Delta$  (Theorem 2) and  $\hat{\Delta}$  (Theorem 3)



- (1) the residual matrix  $E = I - M = I - L^\top AL$
- (2) the  $QR$  decomposition for  $E$  with respect to a special inner product
- (3) the pivoting strategy to detect suitable columns of  $E$  to define  $P$

A detailed (and more technical) description of the implementation can be found in [6]. We refer to this paper for numerical comparisons with other algebraic multilevel methods.

To describe the coarsening process, suppose that we have constructed a  $QR$  decomposition with column pivoting of  $E$ ,

$$E \underbrace{[\Pi_1, \Pi_2]}_{\Pi} = \underbrace{[P, W]}_Q \underbrace{\begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}}_R,$$

where  $Q$  is orthogonal in the inner product given by  $M$ , i.e.,  $Q^\top MQ = I$ , then we already have  $P^\top MW = 0$ .  $P$  and  $W$  fulfill the orthogonality condition of Theorems 2 and 3 and thus the quality of  $P$  as coarse grid projection can be measured via the bounds in these two theorems. If we find a suitable permutation of  $E$  such that sensible estimates for the bounds of Theorems 2 and 3 are small, then  $P$  can serve as coarse grid projection. There are several approaches to compute an approximate  $QR$ -decomposition. One possibility is to adapt a  $QR$ -like decomposition as in [21] but other constructions are possible as well. We refer to [5, 6] for a detailed description of this quite technical construction. There is one drawback of choosing  $P$  itself as coarse grid projection: in general  $P$  is not sparse! But Theorems 2 and 3 are independent of the specific choice of the basis in the space that is spanned by

the columns of  $P$ . Since

$$E\Pi_1 = PR_{11},$$

the columns of  $E\Pi_1$  and  $P$  span the same space. In addition  $E\Pi_1$  refers to specific columns of  $E$  and therefore we can expect  $E\Pi_1$  to be sparse! We use this relation for the coarse grid projection and finally substitute

$$P \rightarrow E\Pi_1$$

as coarse grid projection.

Here we concentrate on the pivoting strategy. Beside many technical details in the implementation of a sparse approximate  $QR$  factorization in [6], there some important components that make this approach effective and that are not as issue of the implementation. They are related to the method of determining the pivots and even show up when using a full  $QR$  decomposition. At first glance the best we can do for pivoting would be to locally maximize  $\Delta$  in inequalities (18), (20), to obtain a feasible coarse grid matrix  $\hat{P} = LP$  (or equivalently  $\hat{P} \rightarrow LE\Pi_1$ ) for  $M^{(1)}$  in (6) and for  $M^{(2)}$  in (7). However, for fixed  $r$  there exist  $\binom{n}{r}$  permutations which have to be checked and for any of these choices one has to compute a  $QR$  decomposition of an  $n \times r$  matrix  $E\Pi_1$  to get the corresponding  $\Delta$ . Already for small  $r$  the costs are prohibitively expensive, e.g. for  $r = 2$ ,  $n(n - 1)/2$  possibilities have to be checked. So in practice not more than  $r = 1$  can be achieved in one step. Using  $P^\top MP = I$  we set

$$(26) \quad T = I - PP^\top M$$

and it is easy to see that the  $M$ -orthogonal complement  $W$  of  $P$  is given by

$$(27) \quad W = TE\Pi_2.$$

Using  $T$  from (26), the bounds (18) and (20) in Theorem 2 can be expressed as

$$(28) \quad \frac{1}{\Delta} = \min_{Tx \neq 0} \frac{x^\top T^\top MTx}{x^\top T^\top Tx},$$

and

$$(29) \quad \frac{1}{\Delta} = \min_{Tx \neq 0} \frac{x^\top (M - EME)x}{x^\top T^\top MTx},$$

respectively.

Analogously, for the bounds in Theorem 3 we have

$$(30) \quad \frac{1}{\hat{\Delta}} = \min_{Tx \neq 0} \frac{x^\top T^\top M^2 Tx}{x^\top T^\top MTx},$$

and

$$(31) \quad \frac{1}{\hat{\Delta}} = \min_{Tx \neq 0} \frac{x^\top T^\top (M - EME)Tx}{x^\top T^\top MTx}$$

respectively.

To compute one of these Rayleigh quotients is still not feasible, since this would have to be done in every step of the  $QR$  decomposition. Even more so, if we have already computed  $r$  columns of the  $QR$ -decomposition, then every choice of column  $r + 1$  has to be checked, i.e., if

$$P_r = [p_1, \dots, p_r]$$

are the first  $r$  columns of  $Q$  in the  $QR$ -decomposition, then any choice of

$$T = I - [P_r, p_{r+1}][P_r, p_{r+1}]^\top M$$

has to be checked. This is far too expensive and even if this were possible, there is no guarantee that choosing the locally optimal  $p_{r+1}$  will globally give a good contribution to the prolongation matrix  $P$ .

The easiest simplification would be to replace the minimum over all  $x$  by a specific choice of  $x$ . An obvious candidate would be the eigenvector of  $M$  associated with its smallest eigenvalue. However, as we will demonstrate in the following examples, local optimization does not necessarily lead to a globally good choice.

EXAMPLE 4. Consider the matrix

$$A = \begin{bmatrix} T & -e_n & 0 \\ -e_n^\top & 1 + \alpha & -\alpha e_1^\top \\ 0 & -\alpha e_1 & \alpha T \end{bmatrix}, \text{ where } T = \begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix} \in \mathbb{R}^{n,n}.$$

for  $\alpha = 100$ . This example can be viewed as discretization of the problem  $-(au)' = f$  in  $[0, 1]$  with  $a(x) = 1$  for  $x \in [0, .5]$  and  $a(x) = \alpha$ ,  $x \in [.5, 1]$ . We will use diagonal preconditioning and the exact eigenvector of the preconditioned system  $M$  associated with the smallest eigenvalue. We repeat the coarsening process on every level and switch to Cholesky decomposition, once the  $QR$ -decomposition needs more than 75% of the columns of  $E$  for the prolongation matrix. Since the coarsening process stops at this stage, we denote this situation in Table 1 with “stopped”.

Table 1 (as well as the following tables) shows the size of the original system (resp. the sequence of coarse grid matrices) and the related fill-in of these matrices for all levels. The fill-in is measured by the density of the matrix compared with a full system (100% means that the matrix is full). In Table 1 one can observe how much the system is reduced in each level of the coarsening process. We denote by  $Est_1$  the estimate obtained using (28) with the smallest eigenvector and by  $Est_2$  the estimate obtained from (29).

TABLE 1. Laplace 1-D, Coarsening. Comparison (for each level) of the number of unknowns and the percentage of fill-in (strategy  $Est_1$  versus strategy  $Est_2$ )

Strategy	Levels: system size $r$ , fill-in (%)				
	1	2	3	4	5
$Est_1$	1023	529, 1	527, 1	stopped	—
$Est_2$		513, 1	257, 1	251, 3	stopped

On the coarser levels  $l > 1$  the following happens. The natural choice would be to choose  $P$  as every second column of  $E$ . This can be seen from the fact that  $2P$  exactly corresponds to the prolongation from geometric multigrid (except the first and the last row). This has happened in the first steps for both strategies, but later the strategies choose the remaining nodes near to the coefficient jump in

$[0.5, 1]$  of the underlying analytic problem. In this form the associated multilevel preconditioners performs very badly.

The situation in Example 4 can also be observed in the two-dimensional case, but here the situation is slightly better.

The example has shown that simply to locally optimize the bounds is not enough. This problem even occurs for the exact QR factorization and represents a fundamental point for the pivoting process. In the sequel we will address this problem and suggest a more effective pivoting strategy. One way out of this dilemma is to choose multiple columns of  $E$  at every step. Since  $P$  spans the same space as suitable chosen columns of  $E$ , we have that two columns  $i, j$  of  $P$  or  $E$  are  $M$ -orthogonal, if their distance is greater than 3 in the graph of  $M$ . This can be seen from the fact that  $E, M$  have the same graph and  $E^\top ME$  may have nonzeros elements only for pairs  $(i, j)$  that have a distance less than or equal to 3. Since any two possible choices for  $p_{r+1}$  commute if their distance in the undirected graph of  $M$  is greater than 3, we can choose as many new nodes in step  $r + 1$  as there are nodes with distance 4 or more between each other. Hence, after  $r$  coarse grid nodes have been chosen, we choose the next coarse grid node such that  $\Delta$  in (28) is minimized for all  $T$  of the form

$$T = I - [P_r, p_{r+1}^{(1)}][P_r p_{r+1}^{(1)}]^\top M$$

Then we continue this procedure for every node of distance greater than 3 from node  $r + 1$ .

$$T = I - [P_r, p_{r+1}^{(1)}, p_{r+1}^{(2)}][P_r, p_{r+1}^{(1)}, p_{r+1}^{(2)}]^\top M$$

We repeat this strategy until there exists no new node with distance greater 3 to all selected nodes of step  $r + 1$ .

Consider again Example 4 using this time multiple columns at one step.

EXAMPLE 5. Using multiple columns in Example 4 we obtain the results depicted in Figure 2 that shows the size of the system and the related fill-in for all levels. We denote by  $Est_{1m}$  the estimate obtained using (28) with the smallest eigenvector and by  $Est_{2m}$  the estimate obtained from (29) in combination with multiple chosen columns.

TABLE 2. Laplace 1-D, Coarsening. Comparison (for each level) of the number of unknowns and the fill-in (strategy  $Est_1$  versus strategy  $Est_2$ , both with multiple column choice)

Strategy	Levels: system size $r$ , fill-in (%)						
	1	2	3	4	5	6	7
$Est_{1m}$	1023	512, 1	256, 1	128, 2	63, 5	31, 9	15, 19
$Est_{2m}$		512, 1	256, 1	128, 2	63, 5	31, 9	15, 19

Here the coarsening process constructs exactly the grid that corresponds to the geometric multigrid case. It follows that the preconditioners are optimal, see Table 3.

A similar improvement can be observed in the two-dimensional case.

TABLE 3. Laplace 1-D. Preconditioned CG: number of iteration steps and flops for different preconditioners (none/diagonal/both multilevel preconditioners)

Strategy	preconditioned system							
	$A$		$MA$		$M_l^{(1)}A$		$M_l^{(2)}A$	
$Est_{1m}$	5037	$8.2 \cdot 10^7$	1023	$2.1 \cdot 10^7$	26	$1.0 \cdot 10^6$	11	$8.7 \cdot 10^5$
$Est_{2m}$					26	$1.0 \cdot 10^6$	11	$8.7 \cdot 10^5$

EXAMPLE 6. Finally we use an example from the Harwell–Boeing collection. The matrix we use is

$$A = \begin{bmatrix} D & B & 0 \\ B^\top & D & \ddots \\ \ddots & \ddots & B \\ 0 & B^\top & D \end{bmatrix}, \quad D = \begin{bmatrix} 786432 & 0 \\ 0 & 256 \end{bmatrix}, \quad B = \begin{bmatrix} -393216 & 6144 \\ -6144 & 64 \end{bmatrix},$$

which essentially corresponds to the matrix LANPRO/NOS2 from the Harwell–Boeing collection. We use this matrix for  $n = 190$  (95 diagonal blocks). Although this matrix is not very big, its condition number is very large and it has large positive off-diagonal entries. This matrix is a real challenge for the coarsening process. Here the multiple column based strategies  $Est_{1m}$  and  $Est_{2m}$  strategies construct coarse grids which end up in perfect multilevel scheme while the single column based versions  $Est_1$  and  $Est_2$  fail already during the generation of the coarse grid. Essentially  $Est_1$  and  $Est_2$  start with similar nodes as  $Est_{1m}$  and  $Est_{2m}$  but then they start taking nodes in the neighbourhood of the previously chosen nodes leading to a coarse grid with insufficient reduction. Table 4 shows the coarsening process and Table 5 gives the behaviour of the preconditioned cg method.

TABLE 4. NOS2, Coarsening. Comparison (for each level) of the number of unknowns and the percentage of fill-in (single column strategies  $Est_1$  and  $Est_2$  versus multiple column strategies  $Est_{1m}$ ,  $Est_{2m}$ )

Strategy	Levels: system size $r$ , fill-in (%)				
	1	2	3	4	5
$Est_1$		129, 8	128, 18	stopped	—
$Est_2$	190, 2	125, 8	118, 19	stopped	—
$Est_{1m}$		94, 6	46, 13	22, 26	10, 52
$Est_{2m}$		94, 6	46, 13	22, 26	10, 52

We should point out that the fast convergence of the cg method for the single column strategy is exclusively based on selecting too many nodes and using the Cholesky decomposition for the final system, where the final system has size 128

and 118 which is pretty close to the size of the original system. The multiple column based strategy impressively demonstrates that the coarsening process can do significantly better. An almost perfect multilevel hierarchy with small number of cg steps is generated.

TABLE 5. NOS2. Preconditioned CG: number of iteration steps and flops for different preconditioners (none/diagonal/both multilevel preconditioners)

Strategy	preconditioned system					
	$A$	$MA$	$M_l^{(1)}A$	$M_l^{(2)}A$		
$Est_1$			20	$3.7 \cdot 10^5$	8	$2.4 \cdot 10^5$
$Est_2$			28	$4.9 \cdot 10^5$	12	$3.4 \cdot 10^5$
$Est_{1m}$	1062	$3.9 \cdot 10^6$	540	$2.5 \cdot 10^6$	22	$2.0 \cdot 10^5$
$Est_{2m}$					22	$2.0 \cdot 10^5$
					9	$1.8 \cdot 10^5$
					9	$1.8 \cdot 10^5$

These examples have illustrated that simply using the locally optimal column is not necessarily optimal from a global point of view, while using multiple columns at every step was very successful. Another technique introduced in [6] consists of locally locking the neighbouring nodes of a chosen column node in the graph theoretical sense. This leads to an alternative way for improving the local optimization. Numerical examples with an efficient implementation of an approximate  $QR$  factorization in [6] show that this kind of algebraic multilevel strategy is competitive with other algebraic multigrid methods especially for non-standard situations, i.e. matrices with many positive off-diagonal entries.

## 5. Conclusion

We have derived estimates for the condition number and hence for the convergence rates of algebraic multilevel preconditioners and demonstrated their quality by examples. These estimates can be used in the construction of algebraic multilevel methods. To do this one computes a special  $QR$ -decomposition with column pivoting, where the pivoting strategy is driven by an estimate of the condition number. This process, however, is in general not monotone. Numerical examples show that this process needs to be supplemented with other heuristic techniques to safeguard the local optimization. These techniques often improve the optimization and in many cases they lead to a good approximation of the global optimum.

## References

1. Owe Axelsson, Maya Neytcheva, and Ben Polman, *An application of the bordering method to solve nearly singular systems*, Vestnik Moskovskogo Universiteta, Seria 15, Vychisl. Math. Cybern. **1** (1996), 3–25.
2. Owe Axelsson, Alexander Padiy, and Ben Polman, *Generalized augmented matrix preconditioning approach and its application to iterative solution of ill-conditioned algebraic systems*, Technical report, Katholieke Universiteit Nijmegen, Fakulteit der Wiskunde en Informatica, 1999.
3. Michele Benzi, Carl D. Meyer, and Miroslav Tůma, *A sparse approximate inverse preconditioner for the conjugate gradient method*, SIAM J. Sci. Comput. **17** (1996), 1135–1149.

4. Michele Benzi and Miroslav Tůma, *A sparse approximate inverse preconditioner for nonsymmetric linear systems*, SIAM J. Sci. Comput. **19** (1998), no. 3, 968–994.
5. Matthias Bollhöfer and Volker Mehrmann, *A new approach to algebraic multilevel methods based on sparse approximate inverses*, Preprint SFB393/99-22, TU Chemnitz, Germany, Dep. of Mathematics, August 1999.
6. ———, *Algebraic multilevel methods and sparse approximate inverses*, SIAM J. Matrix Anal. Appl. **24** (2002), no. 1, 191–218.
7. E. Chow and Y. Saad, *Approximate inverse preconditioners for general sparse matrices*, Research Report UMSI 94/101, University of Minnesota, Super Computing Institute, Minneapolis, Minnesota, 1994.
8. Iain S. Duff, Roger G. Grimes, and John G. Lewis, *Sparse matrix test problems*, ACM Trans. Math. Software **15** (1989), 1–14.
9. Roland W. Freund, Gene H. Golub, and Noel M. Nachtigal, *Iterative solution of linear systems*, Acta Numerica (1992), 1–44.
10. Gene H. Golub and Charles F. Van Loan, *Matrix computations*, third ed., The Johns Hopkins University Press, 1996.
11. Anne Greenbaum, *Iterative methods for solving linear systems*, Frontiers in Applied Mathematics, SIAM Publications, 1997.
12. Marcus J. Grote and Thomas Huckle, *Parallel preconditioning with sparse approximate inverses*, SIAM J. Sci. Comput. **18**(3) (1997), 838–853.
13. Wolfgang Hackbusch, *Multigrid methods and applications*, Springer-Verlag, 1985.
14. ———, *Iterative lösung großer schwachbesetzter Gleichungssysteme*, second ed., B.G. Teubner Stuttgart, 1993.
15. Thomas Huckle, *Matrix multilevel methods and preconditioning*, Technical report SFB-Bericht Nr. 342/11/98 A, Technische Universität München, Fakultät für Informatik, 1998.
16. I. E. Kaporin, *New convergence results and preconditioning strategies for the conjugate gradient method*, Numer. Lin. Alg. w. Appl. **1**(2) (1994), 179–210.
17. L.Yu. Kolotilina and A.Yu. Yeremin, *Factorized sparse approximate inverse preconditionings. I. Theory*, SIAM J. Matrix Anal. Appl. **14** (1993), 45–58.
18. Y. Notay, *Using approximate inverses in algebraic multigrid methods*, Numer. Math. **80** (1998), 397–417.
19. Y. Saad, *Iterative methods for sparse linear systems*, PWS Publishing, Boston, 1996.
20. Y. Saad and M.H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput. **7** (1986), 856–869.
21. G.W. Stewart, *Four algorithms for the efficient computation of truncated pivoted QR approximations to a sparse matrix*, Technical report UMIACS TR-98-12 CMSC TR-3875, University of Maryland, Department of Computer Science, 1998, to appear in Numerische Mathematik.
22. R. S. Varga, *Matrix iterative analysis*, Prentice-Hall, Englewood Cliffs, New Jersey, 1962.

INSTITUT FÜR MATHEMATIK, MA 4–5, TECHNISCHE UNIVERSITÄT BERLIN, D-10623 BERLIN,  
GERMANY

*E-mail address:* bolle@math.tu-berlin.de

*URL:* www.math.tu-berlin.de/~bolle

INSTITUT FÜR MATHEMATIK, MA 4–5, TECHNISCHE UNIVERSITÄT BERLIN, D-10623 BERLIN,  
GERMANY

*E-mail address:* mehrmann@math.tu-berlin.de

*URL:* www.math.tu-berlin.de/~mehrmann

## Spectral equivalence and matrix algebra preconditioners for multilevel Toeplitz systems: a negative result

D. Noutsos, S. Serra Capizzano, and P. Vassalos

**ABSTRACT.** In the last two decades a lot of matrix algebra optimal and superlinear preconditioners (those assuring a strong clustering at the unity) have been proposed for the solution of polynomially ill-conditioned Toeplitz linear systems. The corresponding generalizations to multilevel structures do not preserve optimality neither superlinearity (see e.g. [11]). Regarding the notion of superlinearity, it has been recently shown that this is simply impossible (see [15, 17, 18]). Here we propose some ideas and a proof technique for demonstrating that also the spectral equivalence and the essential spectral equivalence (up to a constant number of diverging eigenvalues) are impossible and therefore the search for optimal matrix algebra preconditioners in the multilevel setting cannot be successful.

### 1. Introduction

In the past few years a lot of attention has been paid to the solution of multilevel Toeplitz systems due to the great variety of applications in which these structures occur such as signal processing, image restoration, PDEs, time series (see e.g. [2]). If  $f$  is a complex-valued function of  $d$  variables, integrable on the  $d$ -cube  $Q_d := (0, 2\pi)^d$ , the symbol  $f_{Q_d}$  stands for  $(2\pi)^{-d} \int_{Q_d}$  and  $\langle x, y \rangle = \sum_j x_j \bar{y}_j$  is the usual inner product, then the Fourier coefficients of  $f$ , given by

$$\widehat{f}_j := \int_{Q_d} f(x) e^{-i \langle j, x \rangle} dx, \quad i^2 = -1, \quad j \in \mathbf{Z}^d,$$

are used for building up  $d$ -level Toeplitz matrices generated by  $f$ . More precisely, if  $n = (n_1, \dots, n_d)$  is a  $d$ -index with positive entries, then the symbol  $T_n(f)$  denotes the  $d$ -level Toeplitz matrix of order  $N(n)$  (throughout, we let  $N(n) := \prod_{i=1}^d n_i$ ) constructed according to the rule

$$(1) \quad T_n(f) = \sum_{|j| < n} \widehat{f}_j J_n^{(j)} = \sum_{|j_1| < n_1} \cdots \sum_{|j_d| < n_d} \widehat{f}_{(j_1, \dots, j_d)} J_{n_1}^{(j_1)} \otimes \cdots \otimes J_{n_d}^{(j_d)}.$$

In the above equation,  $\otimes$  denotes tensor product,  $J_m^{(l)}$  denotes the matrix of order  $m$  whose  $(i, j)$  entry equals 1 if  $j - i = l$  and equals zero otherwise, while  $J_n^{(j)}$ , where  $j$  and  $n$  are multiindices, is the tensor products of all  $J_{n_i}^{(j_i)}$  for  $i = 1, \dots, d$ . In other

---

1991 *Mathematics Subject Classification*. Primary 65F10, 15AXX.

words, the  $2m - 1$  matrices  $\{J_m^{(l)}\}$ ,  $l = 0, \pm 1, \dots, \pm(m - 1)$  are the canonical basis of the linear space of  $m \times m$  Toeplitz matrices, and the tensor notation emphasizes the  $d$ -level Toeplitz structure of  $T_n(f)$  and, indeed, the set  $\{J_n^{(j)}\}$  is the canonical basis of the linear space of the  $N(n) \times N(n)$   $d$ -level Toeplitz matrices.

On the other hand, multilevel Toeplitz matrices are not only interesting from the point of view of the applications [2] or from a “pure mathematics” point of view [1, 21], but also from the viewpoint of the complexity theory since the costs of determining the vector  $\mathbf{u} = T_n(f)\mathbf{v}$  for an arbitrary vector  $\mathbf{v}$  is of  $O(N(n) \log N(n))$  arithmetic operations that is the cost of applying a constant number of multilevel Fast Trigonometric/Fourier transforms.

Now let us come back to the applications and let us recall that the main problem is to solve linear systems of the form  $T_n(f)\mathbf{u} = \mathbf{v}$  for a given vector  $\mathbf{v}$  and for a given  $L^1$  symbol  $f$ . Since the matrix vector multiplication can be performed efficiently, a simple but good idea is to solve the considered linear systems by using iterative solvers in which the involved matrices preserve a Toeplitz structure. Some possibilities are the following: conjugate gradient methods, Chebyshev iterations, Jacobi or Richardson methods with or without polynomial or matrix algebra preconditioning (see [7]). Under these assumptions, the total cost for computing  $\mathbf{u}$  within a preassigned accuracy  $\epsilon$ , is  $O(k_n(\epsilon)N(n) \log N(n))$  where  $k_n(\epsilon)$  is the required number of iterations. If  $f$  is strictly positive and bounded or if the closed convex hull of the range of  $f$  is bounded and does not contain the complex zero, then many of the cited iterations are optimal and we have  $k_n(\epsilon) = O(1)$  [14]. The same is true in the case where  $f$  is continuous, nonnegative, with a finite number of zeros of even orders, the number  $d$  of levels equals 1 and we use a preconditioned conjugate gradient (PCG) method [3, 5, 4, 13, 10].

Here we want to consider the same case ( $f$  nonnegative with a finite number of zeros) but in the multilevel setting i.e.  $d > 1$ . The reason of this attention relies on the importance of the considered case since the discretization of elliptic  $d$ -dimensional PDEs by Finite Differences on equispaced grids leads to sequences  $\{T_n(p)\}$  where  $p$  is positive except at  $x = (0, \dots, 0)$  and is a multivariate trigonometric polynomial. A similar situation occurs in the case of image restoration problems where the sequence  $\{T_n(p)\}$  is associated to a polynomial  $p$  which is positive everywhere with the exception of the point  $x = (\pi, \pi)$ .

Unfortunately, no optimal PCG methods are known in this case in the sense that the number of iterations  $k_n(\epsilon)$  is a mildly diverging function of the dimensions  $n$ . In this paper we will show that the search for spectrally equivalent preconditioners cannot be successful in general and indeed we will illustrate a proof technique for obtaining such negative results.

## 2. Tools, definitions and main results

In the following we will restrict our attention to the simpler case where the generating function  $f$  is nonnegative, bivariate ( $d = 2$ ) and has isolated zeros so that the matrices  $T_n(f)$  are positive definite and ill-conditioned. We will consider as case study two multilevel matrix algebras: the two-level  $\tau$  algebra and the two-level circulant algebra.

The two level  $\tau$  algebra is generated by the pair

$$\Theta_{n_1} \otimes I_{n_2}, \quad I_{n_1} \otimes \Theta_{n_2}$$

in the sense that each matrix of the algebra can be expressed as a bivariate polynomial in the variables  $\Theta_{n_1} \otimes I_{n_2}$  and  $I_{n_1} \otimes \Theta_{n_2}$ ; moreover every matrix of the algebra is simultaneously diagonalized by the orthogonal matrix  $Q$  of size  $N(n)$ . Here  $I_m$  denotes the  $m$ -sized identity matrix,

$$\Theta_m = \begin{pmatrix} 0 & 1 & & \\ 1 & 0 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 0 \end{pmatrix}_m,$$

and the columns of matrix  $Q$  are given by  $v_j^{(n_1)} \otimes v_k^{(n_2)}$  where

$$v_s^{(m)} = \sqrt{\frac{2}{m+1}} \left( \sin \left( \frac{sj\pi}{m+1} \right) \right)_{j=1}^{j=m}.$$

Similarly, the two level circulant algebra is generated (in the same sense as before) by the pair

$$Z_{n_1} \otimes I_{n_2}, \quad I_{n_1} \otimes Z_{n_2}$$

and every matrix of the algebra is simultaneously diagonalized by the unitary Fourier matrix  $F$  of size  $N(n)$ . Here

$$Z_m = \begin{pmatrix} 0 & \dots & 0 & 1 \\ 1 & & & 0 \\ & \ddots & & \vdots \\ 0 & & 1 & 0 \end{pmatrix}_m,$$

and the columns of matrix  $F$  are given by  $f_j^{(n_1)} \otimes f_k^{(n_2)}$  where

$$f_s^{(m)} = \sqrt{\frac{1}{m}} \left( e^{i \frac{2(s-1)(j-1)\pi}{m}} \right)_{j=1}^{j=m}.$$

The strong relationships between these algebras and Toeplitz structures, emphasized by the fact that the generators are of Toeplitz type, have been deeply studied. Here we mention a few examples: given  $p$  bivariate complex polynomial we have

$$T_n(p) = C_n(p) + \tilde{T}_n(p)$$

where  $C_n(p)$  is the two level circulant matrix whose eigenvalues are

$$p_{s,t}^c = p \left( \frac{2(s-1)\pi}{n_1}, \frac{2(t-1)\pi}{n_2} \right)$$

and where  $\tilde{T}_n(p)$  is two level Toeplitz matrix of rank proportional to  $n_1 + n_2$ .

For the case of two level  $\tau$  matrices we have to restrict the attention to real valued even polynomials  $p$ . In that case, we have

$$T_n(p) = \tau_n(p) + H_n(p)$$

where  $\tau_n(p)$  is the two level  $\tau$  matrix whose eigenvalues are

$$p_{s,t}^\tau = p \left( \frac{s\pi}{n_1 + 1}, \frac{t\pi}{n_2 + 1} \right)$$

and where  $H_n(p)$  is two level Hankel matrix of rank proportional to  $n_1 + n_2$ . To make these statements clear we give some examples that will be also useful in the following.

**EXAMPLE 2.1.** Let  $p_k(x, y) = (2 - 2 \cos(x))^k + (2 - 2 \cos(y))^k$  with  $k \geq 1$ . If  $k = 1$  then  $T_n(p_1) = \tau_n(p_1)$  while  $T_n(p_1) = C_n(p_1) + \tilde{T}_n(p_1)$  where

$$(2) \quad \tilde{T}_n(p_1) = (J_{n_1}^{(1-n_1)} + J_{n_1}^{(n_1-1)}) \otimes I_{n_2} + I_{n_1} \otimes (J_{n_2}^{(1-n_2)} + J_{n_2}^{(n_2-1)}).$$

Moreover if  $k = 2$  then  $T_n(p_2) = \tau_n(p_2) + H_n(p_2)$  where

$$(3) \quad H_n(p_2) = (E_{1,1}^{(n_1)} + E_{n_1,n_1}^{(n_1)}) \otimes I_{n_2} + I_{n_1} \otimes (E_{1,1}^{(n_2)} + E_{n_2,n_2}^{(n_2)})$$

with  $E_{j,k}^{(m)}$  being the  $m$  sized matrix having zero entries except for  $(E_{j,k}^{(m)})_{j,k} = 1$ .  $\square$

A tool for proving that a PCG method is optimal when the coefficient matrix is  $A_n$  and the preconditioner is  $P_n$  is the spectral equivalence and the essential spectral equivalence.

**DEFINITION 2.1.** Given  $\{A_n\}_n$  and  $\{P_n\}_n$  two sequences of positive definite matrices of increasing size  $d_n$ , we say that they are spectrally equivalent iff all the eigenvalues of  $\{P_n^{-1}A_n\}_n$  belong to a positive interval  $[\alpha, \beta]$  independent of  $n$  with  $0 < \alpha \leq \beta < \infty$ . We say that the sequences  $\{A_n\}_n$  and  $\{P_n\}_n$  are essentially spectrally equivalent iff there is at most a constant number of outliers and they are all bigger than  $\beta$ .

In practice, in terms of Reileigh quotients, the spectral equivalence means that for any nonzero  $v \in \mathbb{C}^{d_n}$  we have

$$\alpha \leq \frac{v^H A_n v}{v^H P_n v} \leq \beta$$

while the essential spectral equivalence is equivalent to the following two conditions: for any nonzero  $v \in \mathbb{C}^{d_n}$  we have

$$\alpha \leq \frac{v^H A_n v}{v^H P_n v}$$

and for any  $\theta_n$  going to infinity, for any subspace  $\mathcal{V}$  of dimension  $\theta_n$  we have

$$(4) \quad \min_{v \in \mathcal{V}, v \neq 0} \frac{v^H A_n v}{v^H P_n v} \leq \beta.$$

In other words, we can say that there exists a constant positive integer  $q$  independent of  $n$  such that, calling  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_{d_n}$  the eigenvalues of  $P_n^{-1}A_n$ , we have  $\lambda_{q+1} \leq \beta$  and possibly the first  $q$  eigenvalues diverging to infinity as  $n$  tends to infinity. In view of the min max characterization

$$\lambda_{q+1} = \max_{\dim \mathcal{V}=q+1} \min_{v \in \mathcal{V}, v \neq 0} \frac{v^H A_n v}{v^H P_n v}$$

it follows that for any subspace  $\mathcal{V}$  of dimension  $q+1$  we must have (4).

**2.1. Negative results: the  $\tau$  case.** We say that two nonnegative functions  $f$  and  $g$  are equivalent if  $cg(x) \leq f(x) \leq Cg(x)$  for some positive constants  $c$  and  $C$  independent of  $x$  and for any  $x$  belonging to the definition domain. We now state the main negative conjecture.

**CONJECTURE 2.1.** Let  $f$  be equivalent to  $p_k(x, y) = (2 - 2\cos(x))^k + (2 - 2\cos(y))^k$  with  $k \geq 2$  and let  $\alpha$  be a fixed positive number independent of  $n$ . Then for any sequence  $\{P_n\}$  with  $P_n \in \tau_n$  and such that

$$(5) \quad \lambda_{\min}(P_n^{-1}T_n(f)) \geq \alpha$$

uniformly with respect to  $n$ , we have

- (a) : the maximal eigenvalue of  $P_n^{-1}T_n(f)$  diverges to infinity (in other words  $\{T_n(f)\}$  does not possess spectrally equivalent preconditioners in the  $\tau_n$  algebra);
- (b) : if  $\Sigma(X)$  denotes the complete set of the eigenvalues of  $X$ , then the number

$$\#\{\lambda(n) \in \Sigma(P_n^{-1}T_n(f)) : \lambda(n) \rightarrow_{n \rightarrow \infty} \infty\}$$

tends to infinity as  $n$  tends to infinity (in other words  $\{T_n(f)\}$  does not possess essentially spectrally equivalent preconditioners in the  $\tau_n$  algebra).

Since the spectral equivalence and the essential spectral equivalence are equivalence relationships, it is evident that the former conjecture holds in general if it is proved for  $f = p_k$  thus reducing the analysis to a multilevel banded case. In order to show the idea, we prove this negative result in the simplest case where  $k = 2$ .

**Proof of Conjecture 2.1:** items (a) and (b) in the specific case where  $k = 2$ .

Let  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_{N(n)}$  be the eigenvalues of  $P_n^{-1}T_n(p_2)$  with  $P_n$  being a two level  $\tau$  matrix. For the sake of notational simplicity we assume that  $n_1 \sim n_2 \sim m$  i.e. the partial dimensions are proportional. We will prove (a) and (b) with the same argument. By contradiction we suppose that  $\{P_n\}$  and  $\{T_n(p_2)\}$  are essentially spectrally equivalent that is there exist positive constants  $q$ ,  $\alpha$  and  $\beta$  independent of  $n$  such that

$$\lambda_{N(n)} \geq \alpha \text{ and } \lambda_{q+1} \leq \beta.$$

Therefore, from the relation  $\lambda_{N(n)} \geq \alpha$  it follows that  $T_n(p_2) \geq \alpha P_n$  where the relation is in the sense of the partial ordering between Hermitian matrices. On the other hand, from wellknown results on the asymptotic spectra of Toeplitz matrices (see [21] and references therein), we infer that  $T_n(f)$ ,  $n_1 \sim n_2 \sim m$ , has  $g(m)$  eigenvalues  $\lambda(m)$  going to zero asymptotically faster than  $m^{-3}$  i.e.  $\lambda(m) = o(m^{-3})$  where  $g(m) \rightarrow \infty$  with the constraint that  $g(m) = o(m^{1/4})$ . Consequently, since  $P_n \leq \frac{1}{\alpha} T_n(f)$  it follows that also  $P_n$  has  $g(m)$  eigenvalues  $\lambda(m)$  going to zero asymptotically faster than  $m^{-3}$ .

In addition, if  $P_n^{-1}T_n(p_2)$  has at most  $q$  eigenvalues bigger than  $\beta$ , since  $T_n(p_2) \geq \tau_n(p_2)$  (see relation (3)), it follows that  $P_n^{-1}\tau_n(p_2)$  has at most  $q$  outliers and then calling  $\lambda_{s,t}$  the eigenvalue of  $P_n$  related to the eigenvector  $v_s^{(n_1)} \otimes v_t^{(n_2)}$ , it follows that

$$\lambda_{s,t} \geq \frac{1}{\beta} p_{s,t}^\tau$$

with  $p_{s,t}^\tau = p\left(\frac{s\pi}{n_1+1}, \frac{t\pi}{n_2+1}\right)$  (see the beginning of this section) and with at most the exception of  $q$  indices. Therefore the eigenvalues of  $P_n$  that are  $o(m^{-3})$  impose the relation  $o(m^{-3}) = \beta \lambda_{s,t} \geq p_{s,t}^\tau$  and finally the pairs  $(s, t)$  must be such that

$$(s/m)^4 + (t/m)^4 = o(m^{-3}) \text{ i.e.}$$

$$(6) \quad s^4 + t^4 = o(m).$$

This means that the subspace  $\mathcal{W}_n$  spanned by the  $\tau$  eigenvectors related to  $o(m^{-3})$  eigenvalues of  $P_n$  has to be contained in

$$\text{span} \left( v_s^{(n_1)} \otimes v_t^{(n_2)} : s^4 + t^4 = o(m) \right).$$

Define now the following set of indices  $\mathcal{T}_q = \{(s_i, t_i) : i = 0, \dots, q, t_0 < t_1 \dots < t_q\} \subset \{1, \dots, n_1\} \times \{1, \dots, n_2\}$ ,  $\widehat{\mathcal{T}}_q = \{(s_i, t_i) : i = 0, \dots, q, s_0 < s_1 \dots < s_q\} \subset \{1, \dots, n_1\} \times \{1, \dots, n_2\}$  and the corresponding subspaces of dimension  $q+1$ :

$$\mathcal{V}[\mathcal{T}_q] = \text{span} \left( v_s^{(n_1)} \otimes v_t^{(n_2)} : (s, t) \in \mathcal{T}_q \right),$$

$$\mathcal{I}_q = \mathcal{T}_q \text{ either } \mathcal{I}_q = \widehat{\mathcal{T}}_q.$$

Now we look for the contradiction. By using relation (3) we infer the following chain of inequalities

$$\begin{aligned} \beta \geq \lambda_{q+1} &= \max_{\dim \mathcal{V}=q+1} \min_{v \in \mathcal{V}, v \neq 0} \frac{v^H T_n(p_2)v}{v^H P_n v} \\ &= \max_{\dim \mathcal{V}=q+1} \min_{v \in \mathcal{V}, v \neq 0} \frac{v^H (\tau_n(p_2) + H_n(p_2))v}{v^H P_n v} \\ &\geq \max_{\dim \mathcal{V}=q+1} \min_{v \in \mathcal{V}, v \neq 0} \frac{v^H H_n(p_2)v}{v^H P_n v} \\ &\geq \max_{\dim \mathcal{V}=q+1} \min_{v \in \mathcal{V}, v \neq 0} \frac{v^H (E_{1,1}^{(n_1)} + E_{n_1, n_1}^{(n_1)}) \otimes I_{n_2} v}{v^H P_n v} \\ &\geq \max_{\dim \mathcal{V}=q+1} \min_{v \in \mathcal{V}, v \neq 0} \frac{v^H E_{1,1}^{(n_1)} \otimes I_{n_2} v}{v^H P_n v} \\ &= \max_{\dim \mathcal{V}=q+1} \min_{v \in \mathcal{V}, v \neq 0} \frac{v^H \text{diag}(I_{n_2}, 0, \dots, 0)v}{v^H P_n v}. \end{aligned}$$

Moreover, from similar arguments, we also have

$$\begin{aligned} \beta \geq \lambda_{q+1} &= \max_{\dim \mathcal{V}=q+1} \min_{v \in \mathcal{V}, v \neq 0} \frac{v^H T_n(p_2)v}{v^H P_n v} \\ &\geq \max_{\dim \mathcal{V}=q+1} \min_{v \in \mathcal{V}, v \neq 0} \frac{v^H I_{n_1} \otimes E_{1,1}^{(n_2)} v}{v^H P_n v} \\ &= \max_{\dim \mathcal{V}=q+1} \min_{v \in \mathcal{V}, v \neq 0} \frac{v^H \text{diag}(E_{1,1}^{(n_2)}, E_{1,1}^{(n_2)}, \dots, E_{1,1}^{(n_2)}) v}{v^H P_n v}, \end{aligned}$$

As a consequence, setting  $\mathcal{V} = \mathcal{V}[\mathcal{T}_q]$  either  $\mathcal{V} = \mathcal{V}[\widehat{\mathcal{T}}_q]$  with the constraint that  $\mathcal{V}$  is contained in the subspace  $\mathcal{W}_n$  where  $\lambda_{s,t} = o(m^{-3})$ , we obtain

$$\beta \geq \frac{2}{n_1 + 1} \sin^2 \left( \frac{\pi}{n_1 + 1} \right) \left( \max_{(s,t) \in \mathcal{T}_q} \lambda_{s,t} \right)^{-1}$$

either

$$\beta \geq \frac{2}{n_2 + 1} \sin^2 \left( \frac{\pi}{n_2 + 1} \right) \left( \max_{(s,t) \in \widehat{\mathcal{T}}_q} \lambda_{s,t} \right)^{-1}$$

Therefore, since  $\max_{(s,t) \in \mathcal{T}_q} \lambda_{s,t} = o(m^{-3})$  either  $\max_{(s,t) \in \widehat{\mathcal{T}}_q} \lambda_{s,t} = o(m^{-3})$  and  $\frac{2}{n_1+1} \sin^2\left(\frac{\pi}{n_1+1}\right) \geq \frac{c}{m^3}$  for some positive  $c$  independent of  $m$ , we deduce that

$$\beta \cdot o(m^{-3}) \geq \frac{c}{m^3}$$

which is a contradiction.  $\bullet$

**2.2. Negative results: the circulant case.** We begin with the main negative conjecture.

**CONJECTURE 2.2.** *Let  $f$  be equivalent to  $p_k(x,y) = (2 - 2\cos(x))^k + (2 - 2\cos(y))^k$  with  $k \geq 1$  and let  $\alpha$  be a fixed positive number independent of  $n$ . Then for any sequence  $\{P_n\}$  with  $P_n$  two level circulant and such that*

$$(7) \quad \lambda_{\min}(P_n^{-1}T_n(f)) \geq \alpha$$

*uniformly with respect to  $n$ , we have*

- (a) : *the maximal eigenvalue of  $P_n^{-1}T_n(f)$  diverges to infinity (in other words  $\{T_n(f)\}$  does not possess spectrally equivalent preconditioners in the two level circulant algebra);*
- (b) : *if  $\Sigma(X)$  denotes the complete set of the eigenvalues of  $X$ , then the number*

$$\#\{\lambda(n) \in \Sigma(P_n^{-1}T_n(f)) : \lambda(n) \rightarrow_{n \rightarrow \infty} \infty\}$$

*tends to infinity as  $n$  tends to infinity (in other words  $\{T_n(f)\}$  does not possess essentially spectrally equivalent preconditioners in the two level circulant algebra).*

As observed for Conjecture 2.1, we can reduce the analysis to  $f = p_k$ .

**Proof of Conjecture 2.2:** item (a) in the specific case where  $k = 1$ .

Let  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_{N(n)}$  be the eigenvalues of  $P_n^{-1}T_n(p_1)$  with  $P_n$  being a two level circulant matrix. For the sake of notational simplicity we assume that  $n_1 \sim n_2 \sim m$  i.e. the partial dimensions are proportional. By contradiction we suppose that  $\{P_n\}$  and  $\{T_n(p_1)\}$  are spectrally equivalent that is there exists positive constants  $q$ ,  $\alpha$  and  $\beta$  independent of  $n$  such that

$$\lambda_{N(n)} \geq \alpha \text{ and } \lambda_1 \leq \beta.$$

We consider the eigenvectors of the two level circulant algebra  $f_s^{(n_1)} \otimes f_t^{(n_2)}$ ,  $(s,t) \in \{1, \dots, n_1\} \times \{1, \dots, n_2\}$  and by making use of the relation (2) we look for a contradiction. Setting  $v = f_s^{(n_1)} \otimes f_t^{(n_2)}$  and calling  $\lambda_{s,t}$  the corresponding eigenvalue of  $P_n$ , we have

$$\begin{aligned} \beta \geq \lambda_1 &\geq \frac{v^H T_n(p_1) v}{v^H P_n v} \\ &= \frac{v^H C_n(p_1) v + v^H \tilde{T}_n(p_1) v}{v^H P_n v} \\ &= \frac{p_{s,t}^c + v^H \tilde{T}_n(p_1) v}{\lambda_{s,t}}, \end{aligned}$$

where

$$\tilde{T}_n(p_1) = (J_{n_1}^{(1-n_1)} + J_{n_1}^{(n_1-1)}) \otimes I_{n_2} + I_{n_1} \otimes (J_{n_2}^{(1-n_2)} + J_{n_2}^{(n_2-1)})$$

and therefore

$$v^H \tilde{T}_n(p_1)v = \frac{1}{n_1} 2 \cos\left(\frac{2\pi(s-1)}{n_1}\right) + \frac{1}{n_2} 2 \cos\left(\frac{2\pi(t-1)}{n_2}\right).$$

Consequently we have

$$\lambda_{s,t} \geq \frac{1}{\beta} \left( p_{s,t}^c + \frac{1}{n_1} 2 \cos\left(\frac{2\pi(s-1)}{n_1}\right) + \frac{1}{n_2} 2 \cos\left(\frac{2\pi(t-1)}{n_2}\right) \right)$$

and finally, by using simple asymptotical expansions and the explicit expression of  $p_{s,t}^c$  and by recalling that  $n_1 \sim n_2 \sim m$ , we deduce that there exists an absolute constant  $c$  independent of  $m$  such that

$$\lambda_{s,t} \geq \frac{c}{\beta m}$$

On the other hand, from the explicit knowledge of the eigenvalues of  $T_n(p_1)$  and by recalling that  $n_1 \sim n_2 \sim m$ , we infer that  $T_n(f)$  has  $g(m)$  eigenvalues  $\lambda(m)$  going to zero asymptotically faster than  $m^{-1}$  i.e.  $\lambda(m) = o(m^{-1})$  where  $g(m) \rightarrow \infty$  with the constraint that  $g(m) = o(m^{1/2})$ . Finally we deduced a contradiction since at least  $g(m)$  eigenvalues of the preconditioned matrix collapse to zero as  $m$  tends to infinity and this cannot happen under the assumption (7). •

**REMARK 2.1.** Both the proofs given here for the two level case work unchanged in an arbitrary number  $d \geq 2$  of levels with  $k = 1$  (circulant class) and  $k = 2$  ( $\tau$  algebra). The extension of the proof of Conjecture 2.2, item (a) to the case of a general  $k > 1$  and with a different position of the zero seems to be easier than the corresponding generalization of Conjecture 2.1 since in the latter we essentially use the nonnegative definiteness of the low rank correction  $H_n(P_2)$  while in the case of a generic  $k$  the correction  $H_n(P_k)$  is no longer definite and indeed has positive and negative eigenvalues. This essentially means that the proof of item (b) can be a difficult task in general while the generalization of the proof of item (a) should be more straightforward having in mind the scheme of the proof in Conjecture 2.2 with  $k = 1$ . However for both the cases there is a philosophical reason for which these negative results should be generalizable: indeed we have proved these negative results for the easiest cases where the degree of ill-conditioning is the lowest possible among polynomial generating functions. Therefore we expect that a worsening of the condition number should lead to an even more negative result.

### 3. Conclusions

From this note and from [15, 18], the following message can be read: the  $d$  level case with  $d \geq 2$  is dramatically different from the scalar Toeplitz case in terms of preconditioning using fast transforms algebras. More precisely in the  $d$  level case with  $d \geq 2$  it is impossible (except for rare exceptions) to find superlinear and/or (essentially) spectrally equivalent preconditioners. Therefore the only techniques for which the optimality can be proved are those based on multilevel band Toeplitz preconditioning and approximation theory techniques [12, 8, 16] (see also [9] for a new preconditioning scheme). However it should be mentioned that the problem of solving optimally those multilevel banded systems is still a difficult problem that

has been solved (both theoretically and practically) only in some cases with the help of a multigrid strategy [6, 20, 19]. In conclusion, the positive message of this note is the invitation for researchers working in this area to pay more attention to the optimality analysis of multigrid strategies for multilevel banded Toeplitz structures.

## References

- [1] A. Böttcher, B. Silbermann, *Introduction to Large Truncated Toeplitz Operators*. Springer, New York, NY, 1998.
- [2] R.H. Chan, M. Ng, “Conjugate gradient method for Toeplitz systems”, *SIAM Rev.*, **38** (1996), 427–482.
- [3] R.H. Chan, “Toeplitz preconditioners for Toeplitz systems with nonnegative generating functions”, *IMA J. Numer. Anal.*, **11** (1991), 333–345.
- [4] F. Di Benedetto, “Analysis of preconditioning techniques for ill-conditioned Toeplitz matrices”, *SIAM J. Sci. Comp.*, **16** (1995), pp. 682–697.
- [5] F. Di Benedetto, G. Fiorentino, S. Serra, “C.G. Preconditioning for Toeplitz Matrices”, *Comp. Math. Appl.*, **25** (1993), pp. 35–45.
- [6] G. Fiorentino, S. Serra, “Multigrid methods for symmetric positive definite block Toeplitz matrices with nonnegative generating functions”, *SIAM J. Sci. Comp.*, **17-4** (1996), pp. 1068–1081.
- [7] G. Golub, C. Van Loan, *Matrix Computations*. The Johns Hopkins University Press, Baltimore, 1983.
- [8] M. Ng, “Band preconditioners for block-Toeplitz–Toeplitz-block systems”, *Linear Algebra Appl.*, **259** (1997), pp. 307–327.
- [9] D. Noutsos, P. Vassalos, “New band Toeplitz preconditioners for ill-conditioned symmetric positive definite Toeplitz systems”, *SIAM J. Matrix Anal. Appl.*, **23-3** (2002), pp. 728–743.
- [10] D. Potts, G. Steidl, “Preconditioners for ill-conditioned Toeplitz systems constructed from positive kernels”, *SIAM J. Sci. Comput.*, **22-5** (2001), pp. 1741–1761.
- [11] D. Potts, G. Steidl, “Preconditioning of Hermitian block-Toeplitz–Toeplitz-block matrices by level 1 preconditioners”, *Contemp. Math.*, **281** (2001), pp. 193–212.
- [12] S. Serra, “Preconditioning strategies for asymptotically ill-conditioned block Toeplitz systems”, *BIT*, **34** (1994), pp. 579–594.
- [13] S. Serra, “Optimal, quasi-optimal and superlinear band-Toeplitz preconditioners for asymptotically ill-conditioned positive definite Toeplitz systems”, *Math. Comp.*, **66** (1997), pp. 651–665.
- [14] S. Serra Capizzano, P. Tilli, “Extreme singular values and eigenvalues of non-Hermitian block Toeplitz matrices”, *J. Comput. Appl. Math.*, **108-1/2** (1999), pp. 113–130.
- [15] S. Serra Capizzano, E. Tyrtyshnikov, “Any circulant-like preconditioner for multilevel matrices is not superlinear”, *SIAM J. Matrix Anal. Appl.*, **22-1** (1999), pp. 431–439.
- [16] S. Serra Capizzano, “Spectral and computational analysis of block Toeplitz matrices having nonnegative definite matrix-valued generating functions”, *BIT*, **39-1** (1999), pp. 152–175.
- [17] S. Serra Capizzano, E. Tyrtyshnikov, “How to prove that a preconditioner can not be superlinear”, *Math. Comp.*, in press.
- [18] S. Serra Capizzano, “Matrix algebra preconditioners for multilevel Toeplitz matrices are not superlinear”, *Linear Algebra Appl.*, **343/344** (2002), pp. 303–319.
- [19] S. Serra Capizzano, “Convergence analysis of two grid methods for elliptic Toeplitz and PDEs matrix sequences”, *Numer. Math.*, [on line version DOI 10.0007/s002110100331 (2001)].
- [20] H. Sun, X. Jin, Q. Chang, “Convergence of the multigrid method for ill-conditioned block Toeplitz systems”, *BIT*, **41** (2001), PP. 179–190.
- [21] H. Widom, *Toeplitz matrices*. In Studies in real and complex analysis, I. Hirshman Jr. Ed., Math. Ass. Amer., 1965.

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF IOANNINA, GREECE  
*E-mail address:* dnoutsos@cc.uoi.gr

DIPARTIMENTO CFM, UNIVERSITÀ DELL'INSUBRIA - SEDE DI COMO, ITALY  
*E-mail address:* serra@mail.dm.unipi.it, stefano.serrac@uninsubria.it

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF IOANNINA, GREECE  
*E-mail address:* pvassal@pythagoras.math.uoi.gr

## Spectral Distribution of Hermitian Toeplitz Matrices Formally Generated by Rational Functions

William F. Trench

**ABSTRACT.** We consider the asymptotic spectral distribution of Hermitian Toeplitz matrices  $\{T_n\}_{n=1}^\infty$  formally generated by a rational function  $h(z) = (f(z)f^*(1/z))/(g(z)g^*(1/z))$ , where the numerator and denominator have no common zeros,  $\deg(f) < \deg(g)$ , and the zeros of  $g$  are in the open punctured disk  $0 < |z| < 1$ . From Szegő's theorem, the eigenvalues of  $\{T_n\}$  are distributed like the values of  $h(e^{i\theta})$  as  $n \rightarrow \infty$  if  $T_n = \{t_{r-s}\}_{r,s=1}^n$ , where  $\{t_\ell\}_{\ell=-\infty}^\infty$  are the coefficients in the Laurent series for  $h$  that converges in an annulus containing the unit circle. We show that if  $\{t_\ell\}_{\ell=-\infty}^\infty$  are the coefficients in certain other formal Laurent series for  $h$ , then there is an integer  $p$  such that all but the  $p$  smallest and  $p$  largest eigenvalues of  $T_n$  are distributed like the values of  $h(e^{i\theta})$  as  $n \rightarrow \infty$ .

### 1. Introduction

If  $P(z) = a_0 + a_1 z + \cdots + a_k z^k$ , then  $P^*(z) = \bar{a}_0 + \bar{a}_1 z + \cdots + \bar{a}_k z^k$ . We consider the spectral distribution of families of Hermitian Toeplitz matrices  $T_n = \{t_{r-s}\}_{r,s=1}^n$ ,  $n \geq 1$ , where  $\{t_\ell\}_{\ell=-\infty}^\infty$  are the coefficients in a formal Laurent expansion of a rational function

$$h(z) = \frac{f(z)f^*(1/z)}{g(z)g^*(1/z)},$$

where

$$g(z) = \prod_{j=1}^k (z - \zeta_j)^{d_j},$$

$\zeta_1, \dots, \zeta_k$  are distinct,  $0 < |\zeta_r| < 1$  ( $1 \leq r \leq k$ ),  $d_1, \dots, d_k$  are positive integers,  $f$  is a polynomial of degree less than  $d_1 + \cdots + d_k$ , and  $f(\zeta_j)f^*(1/\zeta_j) \neq 0$  ( $1 \leq r \leq k$ ). Then  $h$  has a unique convergent Laurent expansion

$$(1) \quad h(z) = \sum_{\ell=-\infty}^{\infty} \tilde{t}_\ell z^\ell, \quad \max_{1 \leq j \leq k} |\zeta_j| < |z| < \min_{1 \leq j \leq k} 1/|\zeta_j|.$$

If  $\alpha$  and  $\beta$  are respectively the minimum and maximum of  $w(\theta) = h(e^{i\theta})$ , then Szegő's distribution theorem [1, pp. 64-5] implies that eigenvalues of the matrices

---

1991 *Mathematics Subject Classification*. Primary: 15A18; Secondary: 15A57.

$\tilde{T}_n = (\tilde{t}_{r-s})_{r,s=1}^n$ ,  $n \geq 1$ , are all in  $[\alpha, \beta]$ , and are distributed like the values of  $w$  as  $n \rightarrow \infty$ ; that is,

$$(2) \quad \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n F(\lambda_i(\tilde{T}_n)) = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(w(\theta)) d\theta \quad \text{if } F \in C[\alpha, \beta].$$

We are interested in the asymptotic spectral distribution of  $2^k - 1$  other families of Hermitian Toeplitz matrices formally generated by  $h$ , to which Szegö's theorem does not apply. To be specific, a partial fraction expansion yields  $h = h_1 + \cdots + h_k$ , with

$$(3) \quad h_j(z) = \sum_{m=0}^{d_j-1} \left( \frac{a_{mj}}{(1 - \bar{\zeta}_j z)^{m+1}} + \frac{(-1)^m b_{mj} \zeta_j^{m+1}}{(z - \zeta_j)^{m+1}} \right),$$

where  $\{a_{mj}\}$  and  $\{b_{mj}\}$  are constants and

$$(4) \quad a_{d_j-1,j} \neq 0, \quad b_{d_j-1,j} \neq 0, \quad 1 \leq j \leq k.$$

Using the expansions

$$(5) \quad \frac{1}{(1 - \bar{\zeta}_j z)^{m+1}} = \sum_{\ell=0}^{\infty} \binom{m+\ell}{m} \bar{\zeta}_j^\ell z^\ell, \quad |z| < 1/|\zeta_j|,$$

and

$$(6) \quad \frac{(-1)^m \zeta_j^{m+1}}{(z - \zeta_j)^{m+1}} = \sum_{\ell=-\infty}^{-1} \binom{m+\ell}{m} \frac{z^\ell}{\zeta_j^\ell}, \quad |z| > |\zeta_j|,$$

for  $0 \leq m \leq d_j - 1$  produces a Laurent series that converges to  $h_j(z)$  for  $|\zeta_j| < |z| < 1/|\zeta_j|$ . We will call this the *convergent expansion* of  $h_j$ . However, using the expansions

$$(7) \quad \frac{1}{(1 - \bar{\zeta}_j z)^{m+1}} = - \sum_{\ell=-\infty}^{-1} \binom{m+\ell}{m} \bar{\zeta}_j^\ell z^\ell, \quad |z| > 1/|\zeta_j|,$$

and

$$(8) \quad \frac{(-1)^m \zeta_j^{m+1}}{(z - \zeta_j)^{m+1}} = - \sum_{\ell=0}^{\infty} \binom{m+\ell}{m} \frac{z^\ell}{\zeta_j^\ell}, \quad |z| < |\zeta_j|,$$

for  $0 \leq m \leq d_j - 1$  produces a formal Laurent series for  $h_j$  that converges nowhere. We will call this the *formal expansion* of  $h_j$ .

Henceforth eigenvalues are numbered in nondecreasing order. We will prove the following theorem.

**THEOREM 1.** *Let  $\{S_0, S_1\}$  be a partition of  $\{1, \dots, k\}$ , with  $S_1 \neq \emptyset$ . For  $1 \leq j \leq k$ , let  $\sum_{\ell=-\infty}^{\infty} t_\ell^{(j)} z^\ell$  be the convergent expansion of  $h_j$  if  $j \in S_0$ , or the formal expansion of  $h_j$  if  $j \in S_1$ . Let  $T_n = (t_{r-s})_{r,s=1}^n$ , where  $t_\ell = \sum_{j=1}^k t_\ell^{(j)}$ , and let*

$$(9) \quad p = \sum_{j \in S_1} d_j.$$

Then

$$\{\lambda_i(T_n)\}_{i=p+1}^{n-p} \subset [\alpha, \beta], \quad n > 2p,$$

and

$$(10) \quad \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=p+1}^{n-p} F(\lambda_i(T_n)) = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(w(\theta)) d\theta \quad \text{if } F \in C[\alpha, \beta].$$

In fact,

$$(11) \quad \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=p+1}^{n-p} |F(\lambda_i(T_n)) - F(\lambda_i(\tilde{T}_n))| = 0 \quad \text{if } F \in C[\alpha, \beta].$$

We proved a similar theorem in [2], concerning the asymptotic spectral distribution of Hermitian Toeplitz matrices of the form

$$T_n = \sum_j c_j K_n(\zeta_j; P_j) \quad (\text{finite sum})$$

where  $c_1, \dots, c_k$  are real,  $\zeta_1, \dots, \zeta_k$  are distinct and nonzero,  $P_1, \dots, P_k$  are monic polynomials with real coefficients, and

$$K_n(\zeta; P) = \left( P(|r-s|) \rho^{|r-s|} e^{i(r-s)\phi} \right)_{r,s=1}^n.$$

## 2. Proof of Theorem 1

We need the following lemmas from [2].

LEMMA 1. Let

$$\gamma_\ell = \sum_{j=1}^m F_j(\ell) z_j^\ell,$$

where  $z_1, z_2, \dots, z_m$  are distinct nonzero complex numbers and  $F_1, F_2, \dots, F_m$  are polynomials with complex coefficients. Define

$$\mu = \sum_{j=1}^m (1 + \deg(F_j)).$$

Let  $\Gamma_n = (\gamma_{r-s})_{r,s=1}^n$ . Then  $\text{rank}(\Gamma_n) = \mu$  if  $n \geq \mu$ .

LEMMA 2. Let

$$\gamma_r = P(r)\zeta^r + P^*(-r)\bar{\zeta}^{-r},$$

where  $P$  is a polynomial of degreee  $d$  and  $|\zeta| \neq 0, 1$ . Then the Hermitian matrix  $\Gamma_n = (\gamma_{r-s})_{r,s=1}^n$  has inertia  $[d+1, n-2d-2, d+1]$  if  $n \geq 2d+2$ .

(In [2] we considered only the case where  $P$  has real coefficients; however the same argument yields the more general result stated here.)

LEMMA 3. Suppose that  $H_n$  is Hermitian and

$$-\infty < \alpha \leq \lambda_i(H_n) \leq \beta < \infty, \quad 1 \leq i \leq n, \quad n \geq 1.$$

Let  $k$  be a positive integer and let  $p$  and  $q$  be nonnegative integers such that  $p+q = k$ . For  $n \geq k$  let  $T_n = H_n + B_n$ , where  $B_n$  is Hermitian and of rank  $k$ , with  $p$  positive and  $q$  negative eigenvalues. Then

$$\{\lambda_i(T_n)\}_{i=q+1}^{n-p} \subset [\alpha, \beta], \quad n > k,$$

and

$$\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=q+1}^{n-p} |F(\lambda_i(T_n)) - F(\lambda_i(H_n))| = 0 \quad \text{if } F \in C[\alpha, \beta].$$

Moreover,

$$(12) \quad \lambda_i(T_n) - \lambda_i(B_n) = O(1), \quad 1 \leq i \leq q,$$

and

$$(13) \quad \lambda_{n-p+j}(T_n) - \lambda_{n-p+j}(B_n) = O(1), \quad 1 \leq j \leq p,$$

as  $n \rightarrow \infty$ .

In Lemma 3,  $\{H_n\}$  and  $\{B_n\}$  need not be Toeplitz matrices. Our proof of Lemma 3 was motivated in part by an observation of Tyrtyshnikov [3], who, to our knowledge, was the first to apply the idea of low rank perturbations to spectral distribution problems.

Let

$$(14) \quad u_j(z) = \sum_{m=0}^{d_j-1} a_{mj} \binom{m+z}{m} \quad \text{and} \quad v_j(z) = \sum_{m=0}^{d_j-1} b_{mj} \binom{m+z}{m}.$$

From (4),  $\deg(u_j) = \deg(v_j) = d_j - 1$ . We first show that

$$(15) \quad v_j(-z) = u_j^*(z), \quad 1 \leq j \leq k.$$

The proof is by contradiction. Suppose that (15) is false. Let  $\Gamma_n = (\gamma_{r-s})_{r,s=1}^n$ , where

$$(16) \quad \gamma_\ell = \sum_{j=1}^k (v_j(-\ell) - u_j^*(\ell)) \zeta_j^\ell.$$

From Lemma 1, there is a positive integer  $\nu$  such that

$$(17) \quad \text{rank}(\Gamma_n) = \nu, \quad n \geq \nu.$$

From (3), (5), (6), and (14), the convergent expansion of  $h_j$  is

$$h_j(z) = \sum_{\ell=0}^{\infty} u_j(\ell) \bar{\zeta}_j^\ell z^\ell + \sum_{\ell=-\infty}^{-1} v_j(\ell) \zeta_j^{-\ell} z^\ell, \quad |\zeta_j| < |z| < 1/|\zeta_j|.$$

Therefore, the coefficients  $\{\tilde{t}_\ell\}$  in (1) are given by

$$(18) \quad \tilde{t}_\ell = \begin{cases} \sum_{j=1}^k u_j(\ell) \bar{\zeta}_j^\ell, & \ell \geq 0, \\ \sum_{j=1}^k v_j(\ell) \zeta_j^{-\ell}, & \ell < 0. \end{cases}$$

Since  $\tilde{t}_{-\ell} = \bar{\tilde{t}}_\ell$ , this and (16) imply that  $\gamma_\ell = 0$  if  $\ell > 0$ . From this and (17), there is a largest nonpositive integer  $\ell_0$  such that  $\gamma_{\ell_0} \neq 0$ . But then  $\text{rank}(\Gamma_n) = n - |\ell_0|$  if  $n > |\ell_0| + 1$ , which contradicts (17). Therefore, (15) is true.

We can now rewrite (18) as

$$(19) \quad \tilde{t}_\ell = \begin{cases} \sum_{j=1}^k u_j(\ell) \bar{\zeta}_j^\ell, & \ell \geq 0, \\ \sum_{j=1}^k u_j^*(-\ell) \zeta_j^{-\ell}, & \ell < 0. \end{cases}$$

From (3), (7), (8), and (14), the formal expansion of  $h_j(z)$  is

$$-\sum_{\ell=0}^{\infty} v_j(\ell) \zeta_j^{-\ell} z^\ell - \sum_{\ell=-\infty}^{-1} u_j(\ell) \bar{\zeta}_j^\ell z^\ell = -\sum_{\ell=0}^{\infty} u_j^*(-\ell) \zeta_j^{-\ell} z^\ell - \sum_{\ell=-\infty}^{-1} u_j(\ell) \bar{\zeta}_j^\ell z^\ell.$$

Therefore,

$$t_\ell = \begin{cases} \sum_{j \in \mathcal{S}_0} u_j(\ell) \bar{\zeta}_j^\ell - \sum_{j \in \mathcal{S}_1} u_j^*(-\ell) \zeta_j^{-\ell}, & \ell \geq 0, \\ \sum_{j \in \mathcal{S}_0} u_j^*(-\ell) \zeta_j^{-\ell} - \sum_{j \in \mathcal{S}_1} u_j(\ell) \bar{\zeta}_j^\ell, & \ell < 0. \end{cases}$$

(Note that  $t_{-\ell} = \bar{t}_\ell$ .) From this and (19),

$$t_\ell - \bar{t}_\ell = - \sum_{j \in \mathcal{S}_1} (u_j(\ell) \bar{\zeta}_j^\ell + u_j^*(-\ell) \zeta_j^{-\ell}), \quad -\infty < \ell < \infty.$$

Now let  $B_n = T_n - \tilde{T}_n$  and  $\mathcal{S}_1 = \{j_1, \dots, j_k\}$ . Then Lemma 1 with  $m = 2k$ ,

$$\{z_1, z_2, \dots, z_{2k}\} = \{\bar{\zeta}_{j_1}, 1/\zeta_1, \dots, \bar{\zeta}_{j_k}, 1/\zeta_{j_k}\},$$

and

$$\{F_1(\ell), \dots, F_{2k}(\ell)\} = \{u_{j_1}(\ell), u_{j_1}^*(-\ell), \dots, u_{j_k}(\ell), u_{j_k}^*(-\ell)\}, \quad -\infty < \ell < \infty,$$

implies that  $\text{rank}(B_n) = 2p$  if  $n \geq 2p$ . (Recall (9) and that  $\deg(u_j) = d_j - 1$ .)

If  $\Gamma_n^{(j)} = (\gamma_{r-s}^{(j)})_{r,s=1}^n$  with

$$\gamma_\ell^{(j)} = u_j(\ell) \bar{\zeta}_j^\ell + u_j^*(-\ell) \zeta_j^{-\ell},$$

then Lemma 2 implies that  $\Gamma_n^{(j)}$  has  $d_j$  positive and  $d_j$  negative eigenvalues if  $n \geq 2d_j$ . Therefore, the quadratic form associated with  $\Gamma_n^{(j)}$  can be written as a sum of squares with  $d_j$  positive and  $d_j$  negative coefficients if  $n \geq 2d_j$ . It follows that  $B_n$  can be written as a sum of squares with  $p$  positive and  $p$  negative coefficients if  $n \geq 2p$ . Therefore, Sylvester's law of inertia implies that  $B_n$  has  $p$  positive and  $p$  negative eigenvalues if  $n \geq 2p$ . Now Lemma 3 with  $q = p$  implies (11). Since (2) and (11) imply (10), this completes the proof of Theorem 1.  $\square$

From (12) with  $q = p$  and (13), the asymptotic behavior of the  $\lambda_i(T_n)$  for  $1 \leq i \leq p$  and  $n - p + 1 \leq i \leq n$  is completely determined by the asymptotic behavior of the  $2p$  nonzero eigenvalues of  $B_n$ . We believe that the latter all tend to  $\pm\infty$  as  $n \rightarrow \infty$ , but we have not been able to prove this.

## References

- [1] U. Grenander and G. Szegö, *Toeplitz Forms and Their Applications*, Univ. of California Press, Berkeley and Los Angeles, 1958.
- [2] W. F. Trench, *Spectral distribution of generalized Kac-Murdock-Szegö matrices*, Lin. Algebra Appl. 347 (2002), 251-273.
- [3] E. E. Tyrtyshnikov, *A unifying approach to some old and new theorems on distribution and clustering*, Lin. Algebra Appl. 232 (1996), 1-43.

*This page intentionally left blank*

## From Toeplitz Matrix Sequences to Zero distribution of Orthogonal Polynomials

Dario Fasino and Stefano Serra Capizzano

**ABSTRACT.** We review some classical results on the zero distribution of orthogonal polynomials on the light of spectral theory for Toeplitz matrix sequences. In particular, we discuss and extend some recent results by A. Kuijlaars and W. Van Assche on orthogonal polynomials with asymptotically periodic and varying recurrence coefficients.

### 1. Introduction

By Favard's theorem, every three-term recurrence relation of the form

$$(1.1) \quad xp_n(x) = a_{n+1}p_{n+1}(x) + b_n p_n(x) + a_n p_{n-1}(x)$$

with  $a_n > 0$  for  $n > 0$ , where  $p_{-1}(x) = 0$  and  $p_0(x)$  is a constant, determines a sequence  $\{p_n(x)\}$  of polynomials orthonormal with respect to some measure on the real line:

$$(1.2) \quad \int_{\mathbb{R}} p_i(x)p_j(x) d\mu(x) = \delta_{i,j}.$$

The aim of this paper is to review some known results on the distribution of the zeros of polynomials  $p_n(x)$  from the standpoint of spectral theory of Toeplitz matrix sequences. Indeed, the zeros of  $p_n(x)$  are the eigenvalues of the Jacobi matrix

$$(1.3) \quad J_n = \begin{pmatrix} b_0 & a_1 & & \\ a_1 & b_1 & \ddots & \\ \ddots & \ddots & a_{n-1} & \\ & a_{n-1} & b_{n-1} & \end{pmatrix}.$$

Hence, any result concerning spectral properties of such matrix sequences translates immediately into a corresponding result about zeros of  $\{p_n(x)\}$ . As we will see in Section 2 and 3, in some cases the matrix (1.3) is close in some sense to a Toeplitz matrix, or to a block-Toeplitz matrix, or belongs to a sequence with a more concealed structure, but possessing some asymptotic spectral properties. Using this approach, in Section 3 we extend some recent results by A. Kuijlaars, W. Van Assche and the second author [3, 4, 16] on the distribution of zeros of polynomial

---

2000 *Mathematics Subject Classification*. Primary 42C05, 15A18.

sequences with varying coefficients. We devote the last section of this paper to the proof of Theorem 3.2, our main result.

We recall from [6] that the asymptotic zero distribution of the polynomials  $p_n(x)$  is related to other relevant informations on the sequence  $\{p_n(x)\}$ ; in particular, the support of the orthogonality measure  $d\mu(x)$  in (1.2) is contained in the closure of the set of all zeros of all  $p_n(x)$ 's.

To start with, recall that Paul Nevai introduced in [6] the following notation: let  $M(a, b)$  denote the class of orthogonal polynomials (or their corresponding measures, by virtue of the above mentioned Favard's theorem) such that

$$\lim_{n \rightarrow \infty} a_n = \frac{a}{2}, \quad \lim_{n \rightarrow \infty} b_n = b.$$

The following fact is well known [6, 15]:

**THEOREM 1.1.** *Let the sequence  $\{p_n(x)\}$  belong to  $M(a, b)$ , and let  $x_{n,1}, \dots, x_{n,n}$  be the zeros of  $p_n(x)$ . Then, for any continuous function  $F$  with bounded support we have*

$$(1.4) \quad \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n F(x_{n,i}) = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(b + a \cos x) dx.$$

In what follows, we denote by  $C_0$  the set of all real-valued, continuous functions with bounded support. If  $X$  is an Hermitian matrix of order  $n$  let  $\lambda_1(X), \dots, \lambda_n(X)$  denote its eigenvalues considered in nonincreasing order. Moreover, for any set  $\mathcal{I} \subset \mathbb{R}^q$ , let  $m(\mathcal{I})$  denote its Lebesgue measure.

**DEFINITION 1.2.** Let  $f$  be a  $k \times k$  Hermitian matrix-valued function with Lebesgue-measurable entries defined on a set  $\mathcal{I} \subset \mathbb{R}^q$  with finite Lebesgue measure,  $m(\mathcal{I}) < +\infty$ . The matrix sequence  $\{A_n\}$ , where  $A_n$  is Hermitian and has order  $n$ , has the *asymptotic spectral distribution*  $f$  if for all  $F \in C_0$  one has

$$\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{j=1}^n F(\lambda_j(A_n)) = \frac{1}{k m(\mathcal{I})} \sum_{i=1}^k \int_{\mathcal{I}} F(\lambda_i(f(x))) dx.$$

In this case we write in short  $\{A_n\} \sim f$ .

In order to quantify the magnitude of perturbations in a given matrix, we will use *Schatten p-norms*, see e.g., [1, Ch. 4]: If  $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n \geq 0$  are the singular values of the matrix  $X$ , then the Schatten  $p$ -norm of  $X$  is defined as

$$\|X\|_p = \begin{cases} \left( \sum_{j=1}^n \sigma_j^p \right)^{1/p} & 1 \leq p < \infty, \\ \sigma_1 & p = \infty. \end{cases}$$

**LEMMA 1.3.** *Let  $\{A_n\}$  and  $\{B_n\}$  be two sequences of  $n \times n$  Hermitian matrices such that for all  $\varepsilon > 0$  and sufficiently large  $n$  there exists a splitting*

$$A_n = B_n + E_n(\varepsilon) + R_n(\varepsilon).$$

*such that  $\|E_n(\varepsilon)\|_p \leq n^{1/p}\varepsilon$ , for some  $p \in [1, \infty]$ , and the rank of  $R_n(\varepsilon)$  is bounded by  $n\varepsilon$ . Then, for every function  $F \in C_0$ , we have*

$$\lim_{n \rightarrow \infty} \frac{1}{n} \left[ \sum_{i=1}^n F(\lambda_i(A_n)) - F(\lambda_i(B_n)) \right] = 0.$$

A proof of the preceding lemma can be found, e.g., in [14, Thm. 3.1] for  $p = 2$  (Frobenius norm). The general case follows from [11, Thm. 4.4]. According to a standard terminology, see e.g., [14], we say that the eigenvalues of  $\{A_n\}$  and  $\{B_n\}$  are *equally distributed*, and we will write  $\{A_n\} \sim \{B_n\}$ . Obviously, if  $\{A_n\} \sim \{B_n\}$  and  $\{B_n\} \sim f$ , then also  $\{A_n\} \sim f$ .

We can see that Theorem 1.1 follows from the above lemma, if we consider as  $A_n$  the matrix  $J_n$  in (1.3) and put  $B_n = J_n[a, b]$ , where

$$J_n[a, b] = \begin{pmatrix} b & a/2 & & \\ a/2 & b & \ddots & \\ & \ddots & \ddots & a/2 \\ & & a/2 & b \end{pmatrix}.$$

Indeed, its characteristic polynomial  $\det(\lambda I - J_n[a, b])$  is a Chebyshev polynomial of the second kind, suitably scaled and shifted to the interval  $[b - a, b + a]$ . Hence the eigenvalues of the matrix  $J_n[a, b]$  are explicitly given by

$$\lambda_i(J_n[a, b]) = b + a \cos(\pi i / (n + 1)) \quad i = 1, \dots, n,$$

and the limit

$$\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n F(\lambda_i(J_n[a, b])) = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(b + a \cos x) dx$$

is readily established. In particular, if  $\{p_n(x)\} \in M(a, b)$ , then the support of  $d\mu(x)$  is  $[b - a, b + a]$ , plus an at most countable set of mass points (this is the content of Blumenthal's theorem, see e.g., [5, 6]).

We observe that the semi-infinite tridiagonal matrix whose  $n$ -th finite section is  $J_n - J_n[a, b]$  is a compact operator on the space  $\ell^2$  of square summable sequences. Because of the above reason, polynomial sequences in  $M(a, b)$  are said to be *compact perturbations* of Chebyshev polynomials, see [5, 6, 8], and the relation  $\{J_n\} \sim \{J_n[a, b]\}$  is a discrete counterpart of Weil's theorem on the essential spectrum of compact perturbations of self-adjoint operators [5].

Another way to look at the above result, which also leads to useful generalizations as we shall see in what follows, exploits the concept of generating function of a sequence of Toeplitz matrices:

**DEFINITION 1.4.** Let  $f$  be any  $k \times k$  Hermitian matrix-valued function defined in  $(-\pi, \pi)$ , such that the Fourier coefficients

$$(1.5) \quad \mathcal{F}_j[f] = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(x) e^{-ijx} dx \in \mathbb{C}^{k \times k}, \quad j \in \mathbb{Z},$$

are well defined. Then, the function  $f$  is called the *generating function* of the sequence of block Toeplitz matrices

$$T_n(f) = \begin{pmatrix} \mathcal{F}_0[f] & \mathcal{F}_1[f] & \cdots & \mathcal{F}_{n-1}[f] \\ \mathcal{F}_{-1}[f] & \mathcal{F}_0[f] & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathcal{F}_1[f] \\ \mathcal{F}_{1-n}[f] & \cdots & \mathcal{F}_{-1}[f] & \mathcal{F}_0[f] \end{pmatrix} \in \mathbb{C}^{nk \times nk}.$$

Observe that the matrix  $T_n(f)$  is Hermitian, and the index  $n$  here denotes the block order. Then, the limit relation (1.4) is a consequence of the following fact [12, 13, 14]:

**LEMMA 1.5.** *If  $f$  is any function fulfilling the hypotheses of Definition 2, and  $f$  is absolutely integrable, i.e.,*

$$\int_{-\pi}^{\pi} \|f(x)\| dx < +\infty,$$

where  $\|\cdot\|$  is any matrix norm in  $\mathbb{C}^{k \times k}$  (recall that in a finite dimension space all norms are equivalent), then  $\{T_n(f)\} \sim f$  in the sense of Definition 1.2.

Indeed, if we consider the function  $f_{a,b}(x) = b + a \cos x$ , we have  $J_n[a, b] = T_n(f_{a,b})$ , hence  $\{J_n[a, b]\} \sim f_{a,b}$  by Lemma 1.5 with  $k = 1$ . Furthermore,  $\{J_n\} \sim \{J_n[a, b]\}$  by Lemma 1.3 with  $p = \infty$ , and Theorem 1.1 follows since we obtain  $\{J_n\} \sim f_{a,b}$ .

## 2. Asymptotically periodic coefficients

In this section we generalize the class  $M(a, b)$  by considering sequences of coefficients that are asymptotically periodic. Let  $k$  be a fixed positive integer; for any two real vectors of order  $k$ ,  $\mathbf{a} = (a^{(0)}, a^{(1)}, \dots, a^{(k-1)})$  and  $\mathbf{b} = (b^{(0)}, b^{(1)}, \dots, b^{(k-1)})$ , with  $a^{(i)} \geq 0$ , let  $M(\mathbf{a}, \mathbf{b})$  denote the set of all sequences of orthogonal polynomials (or their orthogonality measures, equivalently) defined by the three-term recurrence relations (1.1), where

$$(2.1) \quad \lim_{m \rightarrow \infty} a_{km+j} = \frac{a^{(j)}}{2}, \quad \lim_{m \rightarrow \infty} b_{km+j} = b^{(j)}, \quad j = 0, \dots, k-1.$$

Orthogonal polynomial sequences whose coefficients satisfy the above periodicity assumptions were investigated extensively by Geronimo and Van Assche; a review of some properties of polynomial sequences in this class can be found e.g., in [2]. Clearly, the case where  $k = 1$  coincides with the class  $M(a, b)$ , where  $a = a^{(0)}$  and  $b = b^{(0)}$ .

Consider the tridiagonal matrix  $J_n[\mathbf{a}, \mathbf{b}]$  as in (1.3) whose coefficients are  $a_j = a^{(j \bmod k)}/2$  and  $b_j = b^{(j \bmod k)}$ . Furthermore, let  $f_{\mathbf{a}, \mathbf{b}}(x)$  be the Hermitian matrix-valued trigonometric polynomial

$$(2.2) \quad f_{\mathbf{a}, \mathbf{b}}(x) = J_k[\mathbf{a}, \mathbf{b}] + \frac{1}{2} \begin{pmatrix} 0 & \cdots & 0 & a^{(0)} e^{-ix} \\ \vdots & \cdots & 0 & 0 \\ 0 & 0 & \cdots & \vdots \\ a^{(0)} e^{ix} & 0 & \cdots & 0 \end{pmatrix}.$$

If  $k = 1$ , define  $f_{\mathbf{a}, \mathbf{b}}(x) = b^{(0)} + a^{(0)} \cos x$ . Then, if  $k$  divides  $n$ , we can consider  $J_n[\mathbf{a}, \mathbf{b}]$  as a block Toeplitz matrix whose generating function is  $f_{\mathbf{a}, \mathbf{b}}(t)$ ,

$$J_n[\mathbf{a}, \mathbf{b}] = \begin{pmatrix} \mathcal{F}_0[f_{\mathbf{a}, \mathbf{b}}] & \mathcal{F}_1[f_{\mathbf{a}, \mathbf{b}}] & & O \\ \mathcal{F}_{-1}[f_{\mathbf{a}, \mathbf{b}}] & \mathcal{F}_0[f_{\mathbf{a}, \mathbf{b}}] & \ddots & \\ & \ddots & \ddots & \mathcal{F}_1[f_{\mathbf{a}, \mathbf{b}}] \\ O & & \mathcal{F}_{-1}[f_{\mathbf{a}, \mathbf{b}}] & \mathcal{F}_0[f_{\mathbf{a}, \mathbf{b}}] \end{pmatrix} = T_{n/k}(f_{\mathbf{a}, \mathbf{b}}).$$

In the general case, when  $k$  does not divide  $n$ , the last few rows and columns are missing from  $J_n[\mathbf{a}, \mathbf{b}]$ . This fact does not alter the asymptotic spectral distribution of  $J_n[\mathbf{a}, \mathbf{b}]$ , by virtue of Lemma 1.3. We obtain almost immediately the following result:

**THEOREM 2.1.** *Let  $\{p_n(x)\}$  belong to  $M(\mathbf{a}, \mathbf{b})$ , let  $x_{n,1}, \dots, x_{n,n}$  be the zeros of  $p_n(x)$ , and let  $f_{\mathbf{a}, \mathbf{b}}(t)$  be as in (2.2). Then, for every  $F \in C_0$ , we have*

$$\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{j=1}^n F(x_{n,j}) = \frac{1}{2k\pi} \sum_{i=1}^k \int_{-\pi}^{\pi} F(\lambda_i(f_{\mathbf{a}, \mathbf{b}}(x))) dx$$

or, equivalently,  $\{J_n\} \sim f_{\mathbf{a}, \mathbf{b}}$ .

Indeed, we have  $\{J_n[\mathbf{a}, \mathbf{b}]\} \sim f_{\mathbf{a}, \mathbf{b}}$  by Lemma 3 with  $k = 1$ , and  $\{J_n\} \sim \{J_n[\mathbf{a}, \mathbf{b}]\}$  by Lemma 2 with  $p = \infty$ . As a further consequence, the zeros of the polynomials  $p_n(x)$  are dense in the set

$$(2.3) \quad S = \bigcup_{i=1}^k \text{Range}(\lambda_i(f_{\mathbf{a}, \mathbf{b}})).$$

That set contains the support of the measure of orthogonality  $d\mu(x)$  associated to  $\{p_n(x)\}$ , with the possible exception of at most a denumerable set of point masses.

Owing to the special structure of the matrix (2.2), it is possible to analyze the dependence of its eigenvalues on  $x$  and hence characterize the set  $S$  in (2.3) as follows:

**THEOREM 2.2.** *For  $i = 1, \dots, k$  let*

$$\lambda_i^- = \min\{\lambda_i(f_{\mathbf{a}, \mathbf{b}}(0)), \lambda_i(f_{\mathbf{a}, \mathbf{b}}(\pi))\}, \quad \lambda_i^+ = \max\{\lambda_i(f_{\mathbf{a}, \mathbf{b}}(0)), \lambda_i(f_{\mathbf{a}, \mathbf{b}}(\pi))\}.$$

*Then, for the set  $S$  in (2.3) we have*

$$S = \bigcup_{i=1}^k [\lambda_i^-, \lambda_i^+].$$

**PROOF.** Let  $q(\lambda; x) = \det(f_{\mathbf{a}, \mathbf{b}}(x) - \lambda I)$ , and

$$J_{i,j} = \begin{pmatrix} b^{(i)} & \frac{a^{(i+1)}}{2} & & \\ \frac{a^{(i+1)}}{2} & b^{(i+1)} & \ddots & \\ & \ddots & \ddots & \frac{a^{(j)}}{2} \\ & & \frac{a^{(j)}}{2} & b^{(j)} \end{pmatrix},$$

for  $0 \leq i < j \leq k-1$ . By applying the Laplace's rule to the last row of  $f_{\mathbf{a}, \mathbf{b}}(t) - \lambda I$ , we obtain after simple computations

$$\begin{aligned} q(\lambda; x) &= (b^{(k-1)} - \lambda) \det(J_{0,k-2} - \lambda I) - \frac{a^{(k-1)}{}^2}{4} \det(J_{0,k-3} - \lambda I) + \\ &\quad + \frac{(-1)^k a^{(0)}{}^2}{4} \det(J_{1,k-2} - \lambda I) - (-1)^k (e^{ix} + e^{-ix}) \prod_{j=0}^{k-1} \frac{a^{(j)}}{2} \\ &= p(\lambda) - c \cos x, \end{aligned}$$

where  $p(\lambda)$  is a suitable polynomial of degree  $k$  and  $c = (-1)^k a^{(0)} \cdots a^{(k-1)} / 2^{k-1} \neq 0$ . Since  $f_{\mathbf{a}, \mathbf{b}}(x)$  is Hermitian, its eigenvalues are real. Hence, for all  $x$ , the equation

$p(\lambda) = c \cos x$  has  $k$  real roots, counting multiplicities; in particular, if  $\bar{\lambda}$  is a stationary point of  $p(\lambda)$ , then  $|p(\bar{\lambda})| \leq c$ . Hence, when  $x$  varies between 0 and  $\pi$ , the eigenvalue  $\lambda_i(f_{\mathbf{a},\mathbf{b}}(x))$  varies monotonically between  $\lambda_i(f_{\mathbf{a},\mathbf{b}}(0))$  and  $\lambda_i(f_{\mathbf{a},\mathbf{b}}(\pi))$ .  $\square$

We remark that the above characterization of the support of  $d\mu(x)$  differs from the one given e.g. in Theorems 13 and 14 of [5] (see also [16]), which is stated in terms of continued fractions. Finally, we observe that the claims in the preceding two theorems are left unchanged by any cyclic reordering of the vectors  $\mathbf{a}$  and  $\mathbf{b}$ ; indeed, if  $P$  is any cyclic permutation matrix, then the matrices  $f_{\mathbf{a},\mathbf{b}}(x)$  and  $f_{\mathbf{a}P,\mathbf{b}P}(x)$  constructed as in (2.2) are similar. For example, if  $P$  is the permutation matrix such that  $\mathbf{a}P = (a^{(1)}, \dots, a^{(k-1)}, a^{(0)})$ , then

$$P^T \begin{pmatrix} e^{ix} & & O \\ & 1 & \\ & & \ddots \\ O & & 1 \end{pmatrix} f_{\mathbf{a},\mathbf{b}}(x) \begin{pmatrix} e^{-ix} & & O \\ & 1 & \\ & & \ddots \\ O & & 1 \end{pmatrix} P = f_{\mathbf{a}P,\mathbf{b}P}(x).$$

### 3. Varying coefficients

From here on, we consider the case where the recurrence coefficients  $a_n, b_n$  depend on an extra parameter  $N$ , thus taking the form  $a_{n,N}, b_{n,N}$ . Denote by  $\{p_{n,N}(x)\}$  the corresponding doubly indexed sequence of polynomials, obtained by the initial conditions  $p_{-1,N}(x) = 0$  and  $p_{0,N}(x)$  set to any constant, and by  $x_{n,j}^N$  the corresponding zeros, for  $j = 1, \dots, n$ . In particular, we address the case where the value of  $N = N(n)$  depends on the degree  $n$  in such a way that there exists the limit

$$(3.1) \quad \lim_{n \rightarrow \infty} n/N(n) = t, \quad 0 < t < +\infty.$$

Such a situation may occur when rescaling unbounded sequences  $a_n, b_n$ , as with Stieltjes-Carlitz polynomials [15, 16], and is of interest in those cases where  $a_{n,N}$  and  $b_{n,N}$  are exactly samples of two functions,  $a_{n,N} = a(n/N)$  and  $b_{n,N} = b(n/N)$ , see examples given in [3, 4]. A complementary approach, where the parameter  $N$  affects the measure defining the orthogonality conditions (1.2), was addressed by various authors, see e.g., Proposition 1.3 in [4] and references therein.

Following [3, 4, 16], we use the shorthand

$$\lim_{n/N \rightarrow t} X_{n,N} = X$$

to indicate that

$$\lim_{n \rightarrow \infty} X_{n,N(n)} = X$$

is true whenever  $N(n)$  is any index sequence fulfilling (3.1). With the help of asymptotic estimates on the ratio  $p_{n,N}(x)/p_{n-k,N}(x)$  and potential-theoretic arguments, Van Assche identified the asymptotic zero distribution of the corresponding orthogonal polynomials in the case where the coefficients  $a_{n,N}$  and  $b_{n,N}$  behave like a uniform sampling of continuous functions, see [16, Thm. 2].

Here we want to extend this analysis under more general hypotheses and, more precisely, under the weaker assumption that the coefficients  $a_{n,N}$  and  $b_{n,N}$  behave like a certain “uniform discretization” of two measurable functions in a sense that

will be precised in the forthcoming definition. In what follows,  $[x]$  denotes the largest integer less than or equal to  $x$ .

**DEFINITION 3.1.** Let  $\alpha(s), \beta(s) : \mathbb{R} \mapsto \mathbb{R}^k$  be two given measurable functions,  $\alpha(s) = (\alpha_0(s), \dots, \alpha_{k-1}(s))$  and  $\beta(s) = (\beta_0(s), \dots, \beta_{k-1}(s))$ ; let  $M(\alpha, \beta)$  the class of all doubly indexed sequences of polynomials  $\{p_{n,N}(x)\}$  such that, for every  $t > 0$  and every  $\epsilon > 0$ , then for  $j = 0, \dots, k-1$  we have:

$$\lim_{N \rightarrow \infty} m\{s \in [0, t] : |a_{k[sN]+j, N} - \alpha_j(s)/2| \geq \epsilon\} = 0$$

$$\lim_{N \rightarrow \infty} m\{s \in [0, t] : |b_{k[sN]+j, N} - \beta_j(s)| \geq \epsilon\} = 0.$$

The assumptions in the preceding definition essentially say that, the functions  $\alpha_{N,j}(s) = a_{k[sN]+j, N}$  and  $\beta_{N,j}(s) = b_{k[sN]+j, N}$  converge in measure to  $\alpha_j(s)/2$  and  $\beta_j(s)$ , respectively, when  $N \rightarrow \infty$ , for  $j = 0, \dots, k-1$ . In particular, the above hypotheses are fulfilled when, for all  $t > 0$ ,

$$\lim_{n/N \rightarrow t} a_{kn+j, N} = \frac{\alpha_j(t)}{2}, \quad \lim_{n/N \rightarrow t} b_{kn+j, N} = \beta_j(t), \quad j = 0, \dots, k-1.$$

Observe that we can naturally “embed” the class  $M(\mathbf{a}, \mathbf{b})$  into  $M(\alpha, \beta)$ : If  $\{p_n(x)\} \in M(\mathbf{a}, \mathbf{b})$ , simply let  $p_{n,N}(x) = p_n(x)$ ; thus  $\{p_{n,N}(x)\} \in M(\alpha, \beta)$  with  $\alpha_j(s) \equiv a^{(j)}$  and  $\beta_j(s) \equiv b^{(j)}$ , for  $j = 0, \dots, k-1$ . Our main result is the following:

**THEOREM 3.2.** Let  $\{p_{n,N}(x)\}$  belong to  $M(\alpha, \beta)$ . Let  $A(s)$ ,  $B(s)$  and  $G(s, x)$  be the  $k \times k$  matrix-valued functions

$$A(s) = \begin{pmatrix} & O \\ & \ddots \\ \alpha_0(s) & \end{pmatrix}, \quad B(s) = \begin{pmatrix} \beta_0(s) & \frac{\alpha_1(s)}{2} & & O \\ \frac{\alpha_1(s)}{2} & \beta_1(s) & \ddots & \\ \ddots & \ddots & \ddots & \frac{\alpha_{k-1}(s)}{2} \\ O & \frac{\alpha_{k-1}(s)}{2} & \beta_{k-1}(s) & \end{pmatrix}$$

and  $G(s, x) = B(s) + \frac{1}{2}[A(s)e^{ix} + A(s)^T e^{-ix}]$ . When  $k = 1$  define  $G(s, x) = \beta(s) + \alpha(s) \cos x$ . Then, for every  $F \in C_0$ , the limit

$$(3.2) \quad \lim_{n/N \rightarrow t} \frac{1}{n} \sum_{j=1}^n F(x_{n,j}^N) = \frac{1}{2\pi kt} \sum_{i=1}^k \int_0^t \int_{-\pi}^{\pi} F(\lambda_i(G(s, x))) dx ds$$

is valid for any  $t > 0$  or, equivalently, for any index sequence  $N(n)$  fulfilling (3.1) we have  $\{J_{n,N(n)}\} \sim G_t$  where  $G_t(s, x) = G(st, x)$  is defined over  $[0, 1] \times [-\pi, \pi]$ .

The previous theorem extends the main results of [3, 4], which consider the case  $k = 1$ , and [16, Thm. 2], which deals with arbitrary  $k$  but for continuous functions  $\alpha(s)$  and  $\beta(s)$ . We note that the corresponding results in [4, 16] are obtained from the above mentioned ratio asymptotics on the polynomial sequences, and the final claims are stated in terms of equilibrium measures of the intervals  $[\alpha_i(s), \beta_i(s)]$ . Finally, in parallel with (2.3), we observe that the zeros of  $p_{n,N(n)}$  are dense in the set

$$\bigcup_{i=1}^k \mathcal{ER}(\lambda_i(G_t)),$$

where the symbol  $\mathcal{ER}$  denotes the *essential range*:

$$y \in \mathcal{ER}(\phi) \iff \forall \varepsilon > 0, \quad m\{x : \phi(x) \in (y - \varepsilon, y + \varepsilon)\} > 0.$$

#### 4. Proof of Theorem 3.2

The zeros of  $p_{n,N}(x)$  are the eigenvalues of the Jacobi matrix

$$J_{n,N} = \begin{pmatrix} b_{0,N} & a_{1,N} & & & \\ a_{1,N} & b_{1,N} & & & \\ & \ddots & \ddots & & \\ & & \ddots & a_{n-1,N} & \\ & & & a_{n-1,N} & b_{n-1,N} \end{pmatrix},$$

that is,  $x_{n,j}^N = \lambda_j(J_{n,N})$ . Hence our theorem essentially concerns the spectral distribution of sequences of such Jacobi matrices. To establish Theorem 3.2, we will construct some auxiliary sequences of matrices which will be denoted  $J_n(\epsilon)$ , whose spectral distribution can be computed explicitly and approximates that of  $J_{n,N}$  to any prescribed accuracy, when we take the limit (3.1).

In this section, we will make use of some rather technical results, borrowed from existing literature. The first one is a trivial consequence of the Lusin's theorem [9]:

**LEMMA 4.1.** *Let  $I \subset \mathbb{R}$  be a bounded interval and  $f : I \mapsto \mathbb{R}^k$  be a measurable function. Then for any  $\epsilon > 0$  there exists a continuous function  $f_\epsilon : I \mapsto \mathbb{R}^k$  such that*

$$m\{x \in I : f(x) \neq f_\epsilon(x)\} \leq \epsilon.$$

As a consequence, for any fixed  $t > 0$  and  $\epsilon > 0$ , we can consider continuous functions  $\alpha_\epsilon(s) = (\alpha_{\epsilon,0}(s), \dots, \alpha_{\epsilon,k-1}(s))$  and  $\beta_\epsilon(s) = (\beta_{\epsilon,0}(s), \dots, \beta_{\epsilon,k-1}(s))$ , defined on  $[0, t]$ , such that

$$\begin{aligned} m\{s \in [0, t] : \alpha(s) \neq \alpha_\epsilon(s)\} &\leq \epsilon \\ m\{s \in [0, t] : \beta(s) \neq \beta_\epsilon(s)\} &\leq \epsilon. \end{aligned}$$

We refrain from mentioning explicitly the dependence on  $t$  of  $\alpha_\epsilon$  and  $\beta_\epsilon$ , as well as in other related variables, since all subsequent results are stated for a fixed (but arbitrary)  $t > 0$ . In fact, throughout this section, let  $t > 0$  be fixed and let  $N(n)$  be any index sequence fulfilling (3.1).

Now, let  $A_\epsilon(x)$  and  $B_\epsilon(x)$  be the  $k \times k$  matrix-valued continuous functions

$$(4.1) \quad A_\epsilon(x) = \begin{pmatrix} & & O \\ & & \\ \alpha_{\epsilon,0}(x) & & \end{pmatrix},$$

$$(4.2) \quad B_\epsilon(x) = \begin{pmatrix} \beta_{\epsilon,0}(x) & \frac{\alpha_{\epsilon,1}(x)}{2} & & O \\ \frac{\alpha_{\epsilon,1}(x)}{2} & \beta_{\epsilon,1}(x) & & \\ & \ddots & \ddots & \frac{\alpha_{\epsilon,k-1}(x)}{2} \\ O & & \frac{\alpha_{\epsilon,k-1}(x)}{2} & \beta_{\epsilon,k-1}(x) \end{pmatrix}.$$

When  $k = 1$ , simply set  $A_\epsilon(x) = \alpha_{\epsilon,0}(x)$  and  $B_\epsilon(x) = \beta_{\epsilon,0}(x)$ . Moreover, when  $k$  divides  $n$ , let  $m = [n/k]$  and

$$J_n(\epsilon) = \begin{pmatrix} B_\epsilon(0) & \frac{1}{2}A_\epsilon\left(\frac{1}{2m}t\right) & & O \\ \frac{1}{2}A_\epsilon\left(\frac{1}{2m}t\right)^T & B_\epsilon\left(\frac{1}{m}t\right) & & \\ & \ddots & \ddots & \frac{1}{2}A_\epsilon\left(\frac{2m-3}{2m}t\right) \\ O & & \frac{1}{2}A_\epsilon\left(\frac{2m-3}{2m}t\right)^T & B_\epsilon\left(\frac{m-1}{m}t\right) \end{pmatrix}.$$

When  $n$  is not a multiple of  $k$ , then we pad with zeros a few trailing rows and columns of the previously defined matrix to match the dimension. Observe that  $J_n(\epsilon)$  depends on  $t$  but not on  $N$ . The structure of these matrices is revealed by the following definition:

**DEFINITION 4.2.** Let  $k, n$  be positive integers. Let  $g$  denote any Hermitian matrix-valued function  $g : [0, 1] \times [-\pi, \pi] \mapsto \mathbb{C}^{k \times k}$  such that:

- for all  $s \in [0, 1]$ ,  $g(s, x)$  is integrable in  $x$ ;
- for all  $x \in [-\pi, \pi]$ ,  $g(s, x)$  is continuous in  $s$ .

Define the operator  $\mathcal{T}_n$  as follows: Let  $m = [n/k]$  and

$$\mathcal{T}_n(g) \in \mathbb{C}^{n \times n}, \quad \mathcal{T}_n(g) = \left( \mathcal{F}_{j-i} \left[ g \left( \frac{i+j-2}{2m}, \cdot \right) \right] \right)_{i,j=1}^m \oplus O,$$

where the dot indicates that Fourier integrals (1.5) are computed with respect to the second variable and  $O$  is a null matrix of order  $n \bmod k$ .

The following matrix classes can be expressed in terms of the operators  $\mathcal{T}_n$ :

- Toeplitz matrices: Let  $f : [-\pi, \pi] \mapsto \mathbb{C}$  be integrable, and let  $g(s, x) = f(x)$ . Then  $\mathcal{T}_n(f) = \mathcal{T}_n(g)$ .
- Sampling matrices (see [13, Sect. 5]): Let  $f : [0, 1] \mapsto \mathbb{R}$  be continuous and  $g(s, x) = f(s)$ . Then  $\mathcal{T}_n(g) = \text{Diag}(f(0), f(1/n), \dots, f((n-1)/n))$ .
- Tridiagonal locally Toeplitz matrices: Let  $f_1, f_2 : [0, 1] \mapsto \mathbb{R}$  be two continuous functions and  $g(s, x) = f_1(s) + 2f_2(s)\cos x$ . Then the matrix

$$\mathcal{T}_n(g) = \begin{pmatrix} f_1(0) & f_2\left(\frac{1}{2n}\right) & & O \\ f_2\left(\frac{1}{2n}\right) & f_1\left(\frac{1}{n}\right) & & \ddots \\ & \ddots & \ddots & f_2\left(\frac{2n-3}{2n}\right) \\ O & & f_2\left(\frac{2n-3}{2n}\right) & f_1\left(\frac{n-1}{n}\right) \end{pmatrix}$$

for  $n = 1, 2, \dots$  is easily seen to belong to a sequence of *locally Toeplitz matrices*, see [13, Sect. 5].

The spectral distribution of matrix sequences built up in terms of the operators  $\mathcal{T}_n$  is described by the following theorem whose proof can be simply obtained by a slight variation of results on locally Toeplitz sequences [12, 13].

**THEOREM 4.3.** Let  $A_n = \mathcal{T}_n(g)$ , where  $g$  fulfills the hypotheses stated in Definition 4.2. Then we have  $\{A_n\} \sim g$ , that is, for all  $F \in \mathcal{C}_0$  one has

$$\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{j=1}^n F(\lambda_j(A_n)) = \frac{1}{2\pi k} \sum_{i=1}^k \int_0^1 \int_{-\pi}^{\pi} F(\lambda_i(g(s, x))) dx ds.$$

If we let

$$G_{t,\epsilon}(s, x) = B_\epsilon(ts) + \frac{1}{2} [e^{ix} A_\epsilon(ts) + e^{-ix} A_\epsilon(ts)^T]$$

where  $A_\epsilon(x)$  and  $B_\epsilon(x)$  are as in (4.1) and (4.2), respectively, then it is rather obvious that  $J_n(\epsilon) = \mathcal{T}_n(G_{t,\epsilon})$ . By virtue of the preceding theorem, we have  $\{J_n(\epsilon)\} \sim G_{t,\epsilon}$ .

Now we prove that the difference  $J_{n,N} - J_n(\epsilon)$  can be decomposed into the sum of two matrices, one having a “small” rank and the other having a “small”

norm. Indeed, let  $R_{n,N}(\epsilon)$  be the matrix whose entries equal those corresponding of  $J_{n,N} - J_n(\epsilon)$  that have a modulus exceeding  $2\epsilon$ , and let  $E_{n,N}(\epsilon)$  be such that

$$(4.3) \quad J_{n,N} - J_n(\epsilon) = E_{n,N}(\epsilon) + R_{n,N}(\epsilon).$$

Since  $E_{n,N}(\epsilon)$  is a tridiagonal matrix whose entries are smaller than  $2\epsilon$ , the spectral norm of the matrix  $E_{n,N}(\epsilon)$  is upper bounded by  $6\epsilon$ . Moreover, we can prove that the rank of  $R_{n,N}(\epsilon)$  does not grow too fast, by finding an appropriate upper bound for the number of its nonzero entries. Indeed, let  $j \in \{0, 1, \dots, k-1\}$  be arbitrary and consider first the diagonal entries in  $J_{n,N}$  and  $J_n(\epsilon)$  whose indices are in the set  $\{ik+j+1, i=0, 1, \dots\}$ . For simplicity, we restrict our attention to the case where  $n$  is a multiple of  $k$  and set  $m = n/k$ . Define the set  $\mathcal{E}_n \equiv \mathcal{E}_n(\epsilon, k, j, N, t) \subset [0, t]$  such that

$$\mathcal{E}_n = \{s \in [0, t] : |b_{[sN]k+j, N} - \beta_{\epsilon, j}(s)| > \epsilon\}.$$

By Definition 3.1 and the construction of  $\beta_\epsilon$ , we have

$$(4.4) \quad \limsup_{n \rightarrow \infty} m(\mathcal{E}_n) \leq \epsilon.$$

Moreover, consider the set  $\mathcal{I}_n \equiv \mathcal{I}_n(\epsilon, k, j, N, t) \subset \{0, 1, \dots, m-1\}$  such that

$$\mathcal{I}_n = \left\{ i \in \{0, 1, \dots, m-1\} : \left[ \frac{i}{m}, \frac{i+1}{m} \right] \not\subseteq \mathcal{E}_n \right\}.$$

Hence, if  $i \in \mathcal{I}_n$  then we have

$$\exists s \in \left[ \frac{i}{m}, \frac{i+1}{m} \right] : |b_{ik+j, N} - \beta_{\epsilon, j}(s)| \leq \epsilon,$$

and since  $\beta_{\epsilon, j}$  is uniformly continuous, for sufficiently large  $n$  we have also

$$|b_{ik+j, N} - \beta_{\epsilon, j}(i/m)| \leq 2\epsilon.$$

Now,  $\beta_{\epsilon, j}(i/m)$  is the  $(ik+j+1)$ -th diagonal entry of  $J_n(\epsilon)$ , hence it follows from the construction of  $R_{n,N}(\epsilon)$  that its corresponding entry is zero. Because of (4.4), the number of elements in  $\mathcal{I}_n$  for large  $n$  must exceed  $m(1-2\epsilon)$ . Therefore, by letting  $j$  vary through 0 to  $k-1$ , we see that the number of nonzero diagonal entries of  $R_{n,N}(\epsilon)$  is upper bounded by  $2n\epsilon$ . A similar argument shows that there is a similar bound for co-diagonal entries of  $R_{n,N}(\epsilon)$ , and consequently also for its rank. According to the following definition, borrowed from [10], we have that, for any infinitesimal sequence  $\{\epsilon_m\}$ ,  $\{J_n(\epsilon_m)\}_n : n \in \mathbb{N}$  is an *approximating class of sequences* for the sequence  $\{J_{n,N(n)}\}$ :

**DEFINITION 4.4.** Let  $\{A_n\}$  be a sequence of Hermitian matrices. Then the doubly indexed sequence  $\{\{B_{n,m}\}_n : m \in \mathbb{N}\}$  is an *approximating class of sequences* for  $\{A_n\}$  if, for every integer  $m > 0$  and sufficiently large  $n$ , every  $A_n$  can be split up into the sum of three Hermitian matrices,

$$A_n = B_{n,m} + E_{n,m} + R_{n,m},$$

such that

$$\|E_{n,m}\|_\infty \leq c_1(m), \quad \text{rank } R_{n,m} \leq c_2(m)n,$$

where  $c_1(m)$  and  $c_2(m)$  are independent on  $n$ ,  $\|\cdot\|_\infty$  denotes the spectral norm (Schatten  $\infty$ -norm), and

$$\lim_{m \rightarrow \infty} c_1(m) = \lim_{m \rightarrow \infty} c_2(m) = 0.$$

We are in position to apply the result in [10, Prop. 2.3], which we adapt here to our notations:

LEMMA 4.5. *Let  $\{A_n\}$  be a sequence of Hermitian matrices and let  $\{B_{n,m}\}_n : m \in \mathbb{N}$  be an approximating class of sequences for  $\{A_n\}$ . Suppose that  $\{B_{n,m}\} \sim f_m$  and that  $f_m$  converges in measure to  $f$  as  $m \rightarrow \infty$ . Then, one has  $\{A_n\} \sim f$ .*

In conclusion, from Theorem 8 we deduce  $\{J_n(\epsilon_m)\} \sim G_{t,\epsilon_m}$  with infinitesimal  $\epsilon_m$ , from (4.3) we infer that  $\{\{J_n(\epsilon_m)\}_m : m \in \mathbb{N}\}$  is an approximating class of sequences for  $\{J_{n,N(n)}\}$ , and moreover,  $G_{t,\epsilon_m}$  converges in measure to  $G_t$  as  $m$  tends to infinity, by construction. Hence, Lemma 4.5 implies that  $\{J_{n,N(n)}\} \sim G_t$ , which is the claim contained in Theorem 3.2.

## References

- [1] R. Bhatia. *Matrix Analysis*. Springer-Verlag, New York 1997.
- [2] J. S. Geronimo and W. Van Assche. Orthogonal polynomials with asymptotically periodic recurrence coefficients. *J. Approx. Theory* 46 (1986), 251–283.
- [3] A. B. J. Kuijlaars and S. Serra Capizzano. Asymptotic zero distribution of orthogonal polynomials with discontinuously varying recurrence coefficients. *J. Approx. Theory* 113 (2001), 127–140.
- [4] A. B. J. Kuijlaars and W. Van Assche. The asymptotic zero distribution of orthogonal polynomials with varying recurrence coefficients. *J. Approx. Theory* 99 (1999), 167–197.
- [5] A. Máté, P. Nevai and W. Van Assche. The supports of measures associated with orthogonal polynomials and the spectra of the related selfadjoint operators. *Rocky Mountain J. Math.* 21 (1991), 501–527.
- [6] P. Nevai. *Orthogonal Polynomials*. Memoirs Amer. Math. Soc., no. 213, Providence, RI, 1979.
- [7] P. Nevai and W. Van Assche. Remarks on the  $(C, -1)$ -summability of the distribution of zeros of orthogonal polynomials. *Proc. Amer. Math. Soc.* 122 (1994), 759–767.
- [8] P. Nevai and W. Van Assche. Compact perturbations of orthogonal polynomials. *Pacific J. Math.* 153 (1992), 163–184.
- [9] W. Rudin. *Real and Complex Analysis*. McGraw Hill, 1986.
- [10] S. Serra Capizzano. Distribution results on the algebra generated by Toeplitz sequences: a finite-dimensional approach. *Linear Algebra Appl.* 328 (2001), 121–130.
- [11] S. Serra Capizzano. Spectral behavior of matrix sequences and discretized boundary value problems. *Linear Algebra Appl.* 337 (2001), 37–78.
- [12] S. Serra Capizzano. Generalized locally Toeplitz sequences: spectral analysis and applications to discretized partial differential equations. *Linear Algebra Appl.*, to appear.
- [13] P. Tilli. Locally Toeplitz sequences: spectral properties and applications. *Linear Algebra Appl.* 278 (1998), 91–120.
- [14] E. E. Tyrtyshnikov. A unifying approach to some old and new theorems on distribution and clustering. *Linear Algebra Appl.* 232 (1996), 1–43.
- [15] W. Van Assche. Asymptotic properties of orthogonal polynomials from their recurrence formula. II. *J. Approx. Theory* 52 (1988), 322–338.
- [16] W. Van Assche. Zero distribution of orthogonal polynomials with asymptotically periodic varying recurrence coefficients. *Self-similar systems (Dubna, 1998)* (V. B. Priezhev and V. P. Spiridonov, Eds.), 392–402, Joint Inst. Nuclear Res., Dubna, 1999.

DIPARTIMENTO DI MATEMATICA E INFORMATICA, UNIVERSITÀ DI UDINE, VIA DELLE SCIENZE 206, 33100 UDINE, ITALY

*E-mail address:* fasino@dimi.uniud.it

DIPARTIMENTO DI CHIMICA, FISICA E MATEMATICA, UNIVERSITÀ DELL'INSUBRIA, VIA VALLEGGIO 11, 22100 COMO, ITALY

*E-mail address:* serra@mail.dm.unipi.it

*E-mail address:* stefano.serrac@uninsubria.it

*This page intentionally left blank*

# On Lie Algebras, Submanifolds and Structured Matrices

Kenneth R. Driessel

**ABSTRACT.** I show that manifolds arise naturally when we consider structured eigen-problems for symmetric matrices. In particular, I show that the space of symmetric matrices is naturally partitioned into a collection  $\mathcal{S}$  of connected submanifolds with the following property: For every symmetric matrix  $A$ , the submanifold in  $\mathcal{S}$  containing  $A$  consists of matrices which have the same eigenvalues as  $A$  and the same staircase structure as  $A$ . I also show that the space of symmetric matrices is naturally partitioned into a collection  $\mathcal{T}$  of connected submanifolds with the following property: For every symmetric matrix  $A$ , the submanifold in  $\mathcal{T}$  containing  $A$  consists of matrices which have the same eigenvalues as  $A$  and the same displacement inertia as  $A$ . I obtain these results by considering appropriate Lie algebras of vector fields on the space of symmetric matrices.

## Introduction.

Consider the following general problem:

**The structured eigen-problem.** *Given a structured symmetric matrix, efficiently compute its eigenvalues.*

Of course, this general problem is a class of specific problems. In particular, different specific meanings for “structured” give different specific instances of this general problem.

Mathematicians only know how to handle a few instances of the structured eigen-problem. For example if “structured” means “diagonal” then the eigen-problem is easy: The eigenvalues of a diagonal matrix appear on its diagonal. If “structured” means “circulant” then the eigen-problem is only slightly harder: Every circulant matrix is diagonalized by the Fourier matrix. (See, for example, Davis [1979] and Demmel [1997] Section 6.7 “Fast Fourier transform”.) If “structured” means “tridiagonal” then the eigen-problem is certainly non-trivial but solutions are available: The QR algorithm effectively diagonalizes tridiagonal symmetric matrices. (See, for example, Demmel [1997] Section 5.3.1 “Tridiagonal

---

2000 *AMS Subject Classification.* Primary 15A18 Eigenvalues, singular values, and eigenvectors.

*Key words and phrases.* Lie algebra, vector field, submanifold, iso-spectral surface, eigenvalue, inertia, staircase matrix, displacement structure, iso-spectral surface, group action, group representation, orbit.

QR iteration”, Section 5.5 “Differential equations and eigenvalue problems”, and references therein.)

Here is an instance of the structured eigen-problem which I believe is an important open problem:

**The Toeplitz Eigen-problem.** *Given a symmetric Toeplitz matrix, efficiently compute its eigenvalues.*

How can we find a solution of a structured eigen-problem? I believe that we should first look for a differential equation that solves the problem. Recall that the QR algorithm is closely related to the dynamical system called the “Toda flow”. (See, for example, Deift, Nanda and Tomei [1983], Symes [1980a, 1980b, 1982], Demmel [1997] and references in these works.) The Toda flow provides a great deal of insight concerning the QR algorithm. I regard the Toda flow as more fundamental than the QR algorithm. (See also Driessel [2000].)

**REMARK.** Actually I do not predict that iso-spectral flows will be directly used to effectively calculate eigenvalues. However, I do hope that the study of such flows will lead to matrix factorizations which do so. The relationship between the Toda flow and QR factorizations is described by Symes [1982], Chu [1984], Watkins [1984], Chu and Norris [1988], and Demmel [1997].

But differential equations live on manifolds. This means that we should search for appropriate manifolds along with our search for appropriate differential equations. In other words, we should first analyze the geometry related to a structured eigen-problem.

In this report we shall study several specific geometric objects related to the problems mentioned above. It is usually easy to see that the surface consisting of the symmetric matrices with given structure and given spectrum is an algebraic surface. But algebraic surfaces can have singularities. (For example, consider the curve in the plane defined by  $y^2 = x^3$ . It has a singularity at the origin.) Manifolds are nicer geometric objects because they have well-defined tangent spaces at all of their points. Our aim here is to see that we can work with manifolds rather than algebraic surfaces. In particular, we want to see that manifolds arise naturally when we consider symmetric matrices which have given eigenvalues and given “staircase structure” (defined below). We also want to see that manifolds arise naturally when we consider symmetric matrices which have given eigenvalues and given “displacement structure” (defined below).

In this report, we shall consider several specific meanings for the word “structure” which occurs in the statement of the structured eigen-problem. First we shall consider “staircase structure” which I now define. Let  $\Delta$  be a set of pairs  $(i, j)$  of integers between 1 and  $n$  which satisfy the following conditions: (1) For  $i = 1, \dots, n$ , the diagonal pair  $(i, i)$  is in  $\Delta$ , and (2) if the  $(i, j)$  is in  $\Delta$  then so is the symmetric pair  $(j, i)$ . We regard  $\Delta$  as a **symmetric pattern of interest** of nonzero entries for matrices. A symmetric pattern of interest  $\Delta$  is a **staircase pattern** if it is “filled in toward the diagonal”, that is, for all  $i > j$ , if  $(i, j)$  is in  $\Delta$  then so is  $(i, j + 1)$  and  $(i - 1, j)$ . In this report, we shall see that the space of symmetric matrices is partitioned into a collection  $\mathcal{S}$  of connected submanifolds with the following property: For every symmetric matrix  $A$ , the submanifold in  $\mathcal{S}$  containing  $A$  consists of matrices which have the same eigenvalues as  $A$  and the same staircase structure as  $A$ . This is the first main result of this report. Note

that this result does *not* say that if symmetric matrices  $A$  and  $B$  have the same eigenvalues and the same staircase structure then they are contained in the same submanifold of  $\mathcal{S}$ .

**REMARK.** Arbenz and Golub [1995] showed that the QR algorithm preserves symmetric staircase patterns and only such sparsity patterns. Ashlock, Driessel and Hentzel [1997] showed that the Toda flow preserves symmetric staircase patterns and only such sparsity patterns. Driessel and Gerisch [2001] consider flows which preserve other sparsity patterns.

**REMARK.** In a letter to me (August, 1986), Carlos Tomei wrote:

... consider... the set of all tridiagonal, symmetric matrices with simple fixed spectrum - is this a manifold? One way to do it is to use the Toda flow itself, as you can see in a paper of mine in the Duke Math. J. (1984). You can also try to verify the hypothesis of the implicit function theorem, but I was surprised to see the amount of tricks I had to devise to go that way. Could you find a simple proof...?

In response to Tomei's challenge, I produced a straight-forward proof of the following result:

**PROPOSITION.** *Let  $A$  be an  $n$ -by- $n$  symmetric, tridiagonal matrix with distinct eigenvalues. Then the surface consisting of all symmetric tridiagonal matrices with the same spectrum as  $A$  is regular and hence a manifold.*

The proof appeared in Driessel [1988]. Unfortunately, the argument that I produced depends very much on the specific setting of tridiagonal matrices. The argument that I give here of the corresponding result for staircase matrices (of which tridiagonal matrices are a specific case) applies much more widely.

We shall also consider “displacement structure”. It is well-known (and easy to see) that every  $n$ -by- $n$  symmetric Toeplitz matrix has displacement inertia  $(1, 1, n - 2)$ . (I will review the definition of displacement inertia later.) Consequently, the study of displacement inertia is closely related to the Toeplitz eigenproblem. (See also Fasino [2001].) In this report, we shall see that the space of symmetric matrices is partitioned into a collection  $\mathcal{T}$  of connected submanifolds with the following property: For every symmetric matrix  $A$ , the submanifold in  $\mathcal{T}$  containing  $A$  consists of matrices which have the same eigenvalues as  $A$  and the same displacement inertia as  $A$ . This is the second main result of this report. Note that this result does *not* say that if symmetric matrices  $A$  and  $B$  have the same eigenvalues and the same displacement inertia then they are contained in the same submanifold of  $\mathcal{T}$ .

Here is a summary of the contents of this report. In the section entitled “Lie algebras of vector fields and submanifolds”, I present a brief review of Nagano’s theorem [1966] concerning submanifolds determined by Lie algebras of vector fields on a manifold. In the main body of this report I shall repeatedly use this theorem to define the submanifolds of interest. This theorem provides an easy way to define such submanifolds.

In the section entitled “A Lie algebra of vector fields associated with eigenvalues and similarity by orthogonal matrices”, I present an explicit description of a Lie algebra of vector fields on the space of symmetric matrices which defines the submanifolds consisting of symmetric matrices with given eigenvalues. In the

section entitled “A Lie algebra of vector fields associated with staircase matrices”, I present an explicit description of a Lie algebra of vector fields on the space of symmetric matrices which defines the submanifolds consisting of the matrices with given staircase structure.

The next section is entitled “A Lie algebra of vector fields associated with inertia and congruence”. In it, I present an explicit description of a Lie algebra of vector fields which defines the submanifolds consisting of the symmetric matrices with given inertia. This section provides the basis for the next section which is entitled “A Lie algebra of vector fields associated with displacement inertia and displaced congruence”. In it, I present an explicit description of a Lie algebra of vector fields which defines the submanifolds consisting of the symmetric matrices with given displacement inertia.

The next section is entitled “On symmetric matrices which have the same eigenvalues and the same staircase structure”. In it, I obtain the first main result of this report. In particular, I show that the space of symmetric matrices is partitioned into a collection  $\mathcal{S}$  of connected submanifolds with the following property: For every symmetric matrix  $A$ , the submanifold of  $\mathcal{S}$  containing  $A$  consists of matrices which have the same eigenvalues as  $A$  and the same staircase structure as  $A$ . I obtain this result by considering the intersection of two Lie algebras of vector fields on the space of symmetric matrices. I also show that this intersection contains a substantial number of vector fields. In particular, I show that real analytic functions provide numerous examples of vector fields in this intersection. (These vector fields are not new. See, for example, Demmel [1997] Section 5.5 “Differential equations and eigenvalue problems” and references therein.)

The final main section is entitled “On symmetric matrices which have the same eigenvalues and the same displacement inertia”. In it, I obtain the second main result of this report. In particular, using the same techniques as the previous section, I show that the space of symmetric matrices is partitioned into a collection  $\mathcal{T}$  of connected submanifolds with the following property: For every symmetric matrix  $A$ , the submanifold of  $\mathcal{T}$  containing  $A$  consists of matrices which have the same eigenvalues as  $A$  and the same displacement inertia as  $A$ . I conjecture that the Lie algebra of vector fields defining this collection  $\mathcal{T}$  of submanifolds contains a substantial number of elements.

All of the proofs in the main body of this report are fairly easy. Often when one adopts an appropriate viewpoint, proofs become easy. My first decision (or viewpoint selection) was to view the structured eigen-problem as a search for an appropriate vector field. (This search is not yet finished.) My second decision (or viewpoint selection) was to adopt Lie algebras of vector fields as basic geometric objects. (My second decision is obviously consistent with my first decision.)

I hope that this report will stimulate further study of the submanifolds associated with structured eigen-problems.

### **Lie algebras of vector fields and submanifolds.**

In this section I present a brief review of T. Nagano’s theorem concerning submanifolds determined by Lie algebras of vector fields on a manifold.

**REMARK.** This theorem appeared in a paper that Nagano published in 1966. R. Hermann published similar results in 1960 and 1962. Hermann uses the phrase “foliation with singularities” to describe this situation. For more on the history of

these results and their relationships see Isidori [1985] and Jurdjevic [1997]. I first learned about these results from Olver [1995]. Recall that Lie's original work was stated in terms of vector fields. See Hawkins [2000].

We work in the category of analytic real manifolds. In particular, "smooth" means "analytic" in this paper.

Let  $M$  be an analytic manifold. I use  $Tan.M$  to denote the tangent bundle of  $M$  and, for a point  $p$  in  $M$ , I use  $Tan.M.p$  to denote the space tangent to  $M$  at  $p$ . Recall that a vector field  $u$  on  $M$  is a smooth map  $u : M \rightarrow Tan.M$  which satisfies  $\forall p \in M, u.p \in Tan.M.p$ . Also recall that we may view any vector field on  $M$  as an operator on the algebra of smooth real-valued functions  $f : M \rightarrow R$ . Using this viewpoint, the **Lie bracket**  $[u, v]$  of two vector fields on  $M$  is usually defined to be the vector field  $[u, v] := u \circ v - v \circ u$ , that is,  $[u, v].f := u(v.f) - v(u.f)$ . The set of vector fields on  $M$  with this operation forms a Lie algebra.

**REMARK.** In all the examples discussed in this report, the "ambient" manifold  $M$  is a vector space  $V$ . Then, for any point  $p$  in  $V$ , the space  $Tan.V.p$  may be identified with the space  $V$  itself and any smooth vector field  $u$  on  $V$  may be viewed as a smooth map from  $V$  into  $V$ . If  $u$  and  $v$  are two such vector fields then their Lie bracket  $[u, v]$  may be defined in terms of derivatives as follows: for any  $p \in V$ ,

$$[u, v].p := Du.p.(v.p) - Dv.p.(u.p).$$

**REMARK.** I often use  $f.x$  or  $fx$  in place of  $f(x)$  to indicate function application. I do so to reduce the number of parentheses. I also use association to the left. For example,  $Du.p.(v.p)$  means evaluate  $D$  at  $u$ , then evaluate  $Du$  at  $p$ , then evaluate  $Du.p$  at the result of evaluating  $v$  at  $p$ .

Let  $N$  be a manifold and let  $i : N \rightarrow M$  be a smooth map. Then the pair  $(N, i : N \rightarrow M)$  is an **(immersed) submanifold** of  $M$  if (1) the map  $i$  is injective, and (2) for all  $p \in N$ , the linear map  $Di.p : Tan.N.p \rightarrow Tan.M.(i.p)$  is injective. Clearly we can always replace  $N$  by a subset of  $M$  (namely, the image of  $i$ ) and then replace  $i$  by the corresponding inclusion map.

Let  $L$  be a vector subspace of the Lie algebra of all vector fields on  $M$ . For a point  $p \in M$ , the **integral element**  $L.p$  of  $L$  at  $p$  is the following set:

$$L.p := \{u.p | u \in L\}.$$

Note that the integral element  $L.p$  is a vector subspace of  $Tan.M.p$ . A submanifold  $N$  of  $M$  is an **integral manifold** of  $L$  if (1)  $N$  is connected, and (2) for all  $p$  in  $N$ ,  $Tan.N.p = L.p$ .

Usually a manifold  $M$  is given and then the tangent bundle  $Tan.M$  is determined. The idea here is to proceed in the other order. First the tangent bundle is given by means of a Lie algebra of vector fields. This bundle then determines the manifold. (Hermann [1960] attributes this idea to Cartan.)

**THEOREM (NAGANO'S SUBMANIFOLD THEOREM).** *Let  $M$  be an analytic manifold. Let  $L$  be a subspace of the Lie algebra of all vector fields on  $M$ . If  $L$  is a Lie subalgebra then  $M$  is the disjoint union of integral manifolds of  $L$ .*

The conclusion of this theorem can be restated as follows: The integral manifolds of  $L$  partition  $M$ . In particular, for every  $p \in M$ , there is a unique maximal

integral submanifold of  $L$  thru  $p$ . I shall use  $\text{Subman}(L, p)$  to denote this maximal integral submanifold.

**REMARK.** The term “involutive” is often used in this context: A set  $L$  of vector fields is **involutive** if  $L$  is closed under the Lie bracket operation.

Note that Nagano’s theorem is closely related to the classical theorem of Frobenius on Lie algebras of vector fields and submanifolds. (See, for example, Olver [1995] for the statement of Frobenius’s theorem and a discussion of the relationship of it to Nagano’s theorem.) One of the hypotheses of Frobenius’s theorem is that all the integral elements must have the same dimension. Unfortunately, this condition is not preserved under intersections of Lie algebras of vector fields. In my approach to the structured eigen-problem, I find it very convenient to use such intersections. In particular, I use such intersections to prove the main results of this report.

### A Lie algebra of vector fields associated with eigenvalues and similarity by orthogonal matrices.

In this section I present an explicit description involving the Lie algebra of skew-symmetric matrices, of a Lie algebra of vector fields on the space of symmetric matrices. By Nagano’s submanifold theorem, this Lie algebra determines a partition of the space of symmetric matrices into integral submanifolds. I shall show that these submanifolds are orbits of the similarity action of the orthogonal group.

Let  $Sym(n)$  denote the space of symmetric  $n$ -by- $n$  matrices: in symbols,

$$Sym(n) := \{X \in R^{n \times n} | X^T = X\}.$$

Let  $Skew(n)$  denote the space of  $n$ -by- $n$  skew-symmetric matrices: in symbols.

$$Skew(n) := \{K \in R^{n \times n} | K^T = -K\}.$$

Let  $j : Sym(n) \rightarrow Skew(n)$  be an analytic map. With  $j$ , we associate a vector field  $\kappa.j$  defined as follows:

$$\kappa.j := Sym(n) \rightarrow Sym(n) : X \mapsto [X, j.X].$$

In particular, note that if  $X$  is symmetric and  $K$  is skew-symmetric then  $[X, K]$  is symmetric. (Here  $[A, B]$  denotes the **commutator** of matrices which is defined by  $[A, B] := AB - BA$ .)

**PROPOSITION.** *The set of all vector fields  $\kappa.j$ , such that  $j : Sym(n) \rightarrow Skew(n)$  is an analytic map, forms a Lie subalgebra of the Lie algebra of all analytic vector fields on the space of symmetric matrices. In particular, if  $j : Sym(n) \rightarrow Skew(n)$  and  $k : Sym(n) \rightarrow Skew(n)$  are analytic maps then  $[\kappa.j, \kappa.k] = \kappa.l$  where  $l := Sym(n) \rightarrow Skew(n)$  is defined by*

$$X \mapsto Dj.X.(\kappa.k.X) - Dk.X.(\kappa.j.X) - [j.X, k.X].$$

**REMARK.** Note that the bracketed pair  $[\kappa.j, \kappa.k]$  is a commutator of vector fields. The other bracketed pair in this proposition is a matrix commutator.

**PROOF.** Let  $L$  denote the set of vector fields defined in the proposition. Note that the map  $\kappa$  is linear. It follows that its image  $L$  is a vector space.

We now compute the commutator  $[\kappa.j, \kappa.k]$ . Note that for any symmetric matrices  $X$  and  $Y$  we have

$$D(\kappa.j).X.Y = [Y, j.X] + [X, D.j.X.Y].$$

Using this formula, we have, for any symmetric matrix  $X$ ,

$$\begin{aligned} [\kappa.j, \kappa.k].X &= D(\kappa.j).X.(\kappa.k.X) - D(\kappa.k).X.(\kappa.j.X) \\ &= [\kappa.k.X, j.X] + [X, D.j.X.(\kappa.k.X)] - [\kappa.j.X, k.X] - [X, D.k.X.(\kappa.j.X)] \\ &= [X, D.j.X.(\kappa.k.X) - D.k.X.(\kappa.j.X)] + [X, [k.X, j.X]]. \end{aligned}$$

Note that the last step used Jacobi's identity for matrix commutators.  $\square$

I shall use *Lie.Spectra* to denote the Lie algebra of vector fields defined in the proposition; in symbols,

$$\text{Lie.Spectra} := \{\kappa.j | j : \text{Sym}(n) \rightarrow \text{Skew}(n) \text{ is analytic}\}.$$

I shall call this set the **Lie algebra of vector fields associated with spectra or eigenvalues**. It follows from Nagano's submanifold theorem that this Lie algebra of vector fields determines a partition of the space of symmetric matrices into integral submanifolds. In fact these submanifolds are orbits of the similarity representation of the orthogonal group on the space of symmetric matrices as I shall now indicate.

Let  $O(n)$  denote the orthogonal group; in symbols,

$$O(n) := \{Q \in R^{n \times n} | QQ^T = I\}.$$

Consider the action  $\alpha$  of  $O(n)$  on the space of symmetric matrices by orthogonal similarity:

$$\alpha := O(n) \times \text{Sym}(n) \rightarrow \text{Sym}(n) : (Q, X) \mapsto QXQ^T.$$

For a symmetric matrix  $X$  we have the orbit of  $X$ ,

$$\alpha.O(n).X := \{QXQ^T | Q \in O(n)\},$$

under this action. By the spectral theorem, this orbit is the **iso-spectral surface** consisting of the symmetric matrices which have the same eigenvalues as  $X$ . Note that diagonal matrices  $\text{diag}(\lambda_1, \dots, \lambda_n)$  provide a canonical form for this group action.

Consider the following map determined by a symmetric matrix  $X$  and the action  $\alpha$ :

$$\alpha^\vee.X := O(n) \rightarrow \text{Sym}(n) : Q \mapsto QXQ^T.$$

Note that the image of this map is the orbit  $\alpha.O(n).X$ . We can differentiate the map  $\alpha^\vee.X$  to get the following linear map:

$$D(\alpha^\vee.X).I = \text{Tan}.O(n).I \rightarrow \text{Tan}.Sym(n).X : K \mapsto [K, X].$$

Recall that the space tangent to the group  $O(n)$  at the identity  $I$  may be identified with the skew-symmetric matrices; in symbols,

$$\text{Tan}.O(n).I = \text{Skew}(n).$$

(See, for example, Curtis [1984].) Also note that the space tangent to the space  $Sym(n)$  at  $X$  may be identified with the space  $Sym(n)$  itself; in symbols,

$$Tan.Sym(n).X = Sym(n).$$

Consequently, we can, and shall, regard the derivative  $D(\alpha^\vee.X).I$  as a linear map from  $Skew(n)$  to  $Sym(n)$ . It is not hard to prove that the space tangent to the iso-spectral surface at  $X$  is the image of this linear map. (Hint: For skew-symmetric  $K$  consider the curve  $t \mapsto exp(tK)$  in  $O(n)$ .) In symbols, we have

$$Tan.(\alpha.O(n).X).X = \{[K, X] | K \in Skew(n)\}.$$

Note that for smooth  $j : Sym(n) \rightarrow Skew(n)$  the vector field  $\kappa.j$  is tangent to these orbits; in symbols, for all symmetric  $X$ ,

$$\kappa.j.X \in Tan.(\alpha.O(n).X).X.$$

In fact, the integral element of the system of vector fields *Lie.Spectra* at a symmetric matrix  $X$  is this tangent space; in symbols,

$$Lie.Spectra.X = \{[K, X] | K \in Skew(n)\}.$$

In particular, note that, for any skew-symmetric  $K$ , we have the constant map  $j := Sym(n) \rightarrow Skew(n) : Y \mapsto K$  and then  $\kappa.j = Sym(n) \rightarrow Sym(n) : X \mapsto [X, K]$ . Thus the orbits of the similarity action of the orthogonal group are the integral submanifolds of the Lie algebra *Lie.Spectra* of vector fields. (In particular, note that the orbits are connected.)

### A Lie algebra of vector fields associated with staircase structure and similarity by upper triangular matrices.

In this section I present an explicit description involving upper triangular matrices, of a Lie algebra of vector fields on the space of symmetric matrices. By Nagano's submanifold theorem, this Lie algebra determines a partition of the space of symmetric matrices into integral submanifolds. I shall show that these submanifolds are orbits of an action of the group of invertible upper triangular matrices.

Let  $upper(n)$  denote the space of upper triangular matrices; in symbols,

$$upper(n) := \{R \in R^{n \times n} | i > j \Rightarrow R(i, j) = 0\}.$$

For a square matrix  $X$ , let  $X_l$ ,  $X_d$  and  $X_u$  denote the strictly lower triangular, diagonal and strictly upper triangular part of  $X$ . Note that the space of square matrices  $R^{n \times n}$  is the direct sum of the space of symmetric matrices and the space of strictly upper triangular matrices; in particular,

$$X = (X_l + X_d + X_l^T) + (X_u - X_l^T).$$

Let  $\sigma$  denote the projection of  $R^{n \times n}$  onto the subspace of symmetric matrices along the subspace of strictly upper triangular matrices; in symbols,

$$\sigma := R^{n \times n} \rightarrow Sym(n) : X \mapsto X_l + X_d + X_l^T.$$

Let  $u : Sym(n) \rightarrow upper(n)$  be an analytic map. With  $u$ , we associate a vector field  $\tau.u$  defined as follows:

$$\tau.u := Sym(n) \rightarrow Sym(n) : X \mapsto \sigma[X, u.X].$$

**PROPOSITION.** *The set of all vector fields  $\tau.u$ , such that the map  $u : \text{Sym}(n) \rightarrow \text{upper}(n)$  is analytic, forms a Lie subalgebra of the Lie algebra of all analytic vector fields on the space of symmetric matrices. In particular, if  $u : \text{Sym}(n) \rightarrow \text{upper}(n)$  and  $v : \text{Sym}(n) \rightarrow \text{upper}(n)$  are analytic maps then  $[\tau.u, \tau.v] = \tau.w$  where  $w := \text{Sym}(n) \rightarrow \text{upper}(n)$  is defined by*

$$X \mapsto Du.X.(\tau.v.X) - Dv.X.(\tau.u.X) - [u.X, v.X].$$

**PROOF.** Let  $L$  denote the set of vector fields defined in the proposition. Since  $\tau$  is linear, it follows that its image  $L$  is a vector space.

We now compute the commutator  $[\tau.u, \tau.v]$ . Note that for any symmetric matrices  $X$  and  $Y$  we have

$$D(\tau.u).X.Y = \sigma[Y, u.X] + \sigma[X, Du.X.Y].$$

Using this formula, we have, for any symmetric matrix  $X$ ,

$$\begin{aligned} [\tau.u, \tau.v].X &= D(\tau.u).X.(\tau.v.X) - D(\tau.v).X.(\tau.u.X) \\ &= \sigma[\tau.v.X, u.X] + \sigma[X, Du.X.(\tau.v.X)] \\ &\quad - \sigma[\tau.u.X, v.X] - \sigma[X, Dv.X.(\tau.u.X)] \\ &= \sigma[X, Du.X.(\tau.v.X) - Dv.X.(\tau.u.X)] \\ &\quad + \sigma[\sigma[X, v.X], u.X] - \sigma[\sigma[X, u.X], v.X]. \end{aligned}$$

To finish the proof, we need to see that, for upper triangular matrices  $U$  and  $V$ ,

$$\sigma[X, [U, V]] = \sigma[\sigma[X, U], V] + \sigma[U, \sigma[X, V]].$$

(Note that this equation looks very much like Jacobi's identity.) This equation follows easily from the following assertion.

*Claim.* For all  $Y \in R^{n \times n}$  and  $R \in \text{upper}(n)$ ,  $\sigma[\sigma Y, R] = \sigma[Y, R]$ .

We have

$$\begin{aligned} \sigma[\sigma Y, R] &= \sigma[Y_l + Y_d, R] + \sigma[Y_l^T, R] \\ &= \sigma[Y_l + Y_d, R] = \sigma[Y_l + Y_d, R] + \sigma[Y_u, R] \\ &= \sigma[Y, R]. \end{aligned}$$

□

I shall use *Lie.Staircase* to denote the Lie algebra of vector fields defined in this proposition; in symbols,

$$\text{Lie.Staircase} := \{\tau.u : \text{Sym}(n) \rightarrow \text{upper}(n) \text{ is analytic}\}.$$

I call this set the **Lie algebra of vector fields associated with staircase structure**. It follows from Nagano's submanifold theorem that this Lie algebra of vector fields determines a partition of the space of symmetric matrices into integral submanifolds. In fact, these submanifolds are orbits of a representation of a matrix group on the space of symmetric matrices as I shall now indicate. I shall also indicate the motivation for the phrase "associated with staircase structure" used above.

Let  $\text{Upper}(n)$  denote the group of invertible  $n$ -by- $n$  upper triangular matrices; in symbols,

$$\text{Upper}(n) := \{U \in Gl(n) | i > j \Rightarrow U(i, j) = 0\}.$$

Consider the following map:

$$\beta := \text{Upper}(n) \times \text{Sym}(n) \rightarrow \text{Sym}(n) : (U, X) \mapsto \sigma(U X U^{-1}).$$

**PROPOSITION.** *The map  $\beta$  is a group representation.*

**PROOF.** That  $\beta$  is a group representation follows easily from the fact that the strictly upper triangular matrices are invariant under the following similarity action:

$$\text{Upper}(n) \times R^{n \times n} \rightarrow R^{n \times n} : (U, X) \mapsto U X U^{-1}.$$

For details see, for example, Driessel and Gerisch [2001].  $\square$

For symmetric matrix  $X$  we have the orbit of  $X$ ,

$$\beta.\text{Upper}(n).X := \{\sigma(U X U^{-1}) : U \in \text{Upper}(n)\}.$$

under the action  $\beta$ . The next proposition indicates a connection between these orbits and staircase structure.

Recall that a symmetric collection  $\Delta$  of pairs  $(i, j)$  is a staircase pattern of interest if it is filled in toward the diagonal. Note that these staircase patterns are closed under intersection. Consequently, every pattern of matrix entries **generates** a staircase pattern, namely, the smallest staircase pattern which includes the given pattern. In particular, with every symmetric matrix  $X$ , we can associate the **staircase pattern of  $X$**  generated by the nonzero entries in  $X$ ; in other words, the staircase pattern of  $X$  is the smallest staircase pattern which includes all of the pairs  $(i, j)$  such that  $X(i, j) \neq 0$ .

**PROPOSITION.** *Let  $X$  be a symmetric matrix. If  $Y$  is any symmetric matrix in the  $\beta$  orbit of  $X$  then  $Y$  has the same staircase pattern as  $X$ .*

**PROOF.** Here are the main ideas of the proof: The staircase pattern of  $X$  is determined by the nonzero entries in  $X$  which are furthest southwest. These entries do not become 0 under the action of  $\beta$ . Value flows toward the north and east under this action.

Here are a few more details. Note that every element of the upper triangular group  $\text{Upper}(n)$  can be written as the product of an invertible diagonal matrix and an upper triangular matrix with ones on the diagonal. We can treat these two kinds of matrices separately since  $\beta$  is a group action.

Let  $D := \text{diag}(d_1, \dots, d_n)$  be an invertible diagonal matrix. Then, for  $i \geq j$ , we have  $\beta.D.X(i, j) = d_i X(i, j) d_j^{-1}$ .

Let  $U$  be an upper triangular matrix with ones on its diagonal. Then, for  $i > j$ , if the entries of  $X$  are zero west and south of  $(i, j)$ , then  $\beta.U.X(i, j) = X(i, j)$ .  $\square$

The following example illustrates the last part of the proof. The entries with \* are “don’t care” entries.

$$\begin{aligned} & \begin{pmatrix} 1 & * & * & * \\ 0 & 1 & * & * \\ 0 & 0 & 1 & * \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} * & 1 & 0 & 0 \\ 1 & * & * & 1 \\ 0 & * & * & * \\ 0 & 1 & * & * \end{pmatrix} \begin{pmatrix} 1 & * & * & * \\ 0 & 1 & * & * \\ 0 & 0 & 1 & * \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} 1 & * & * & * \\ 0 & 1 & * & * \\ 0 & 0 & 1 & * \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} * & * & * & * \\ 1 & * & * & * \\ 0 & * & * & * \\ 0 & 1 & * & * \end{pmatrix} = \begin{pmatrix} * & * & * & * \\ 1 & * & * & * \\ 0 & * & * & * \\ 0 & 1 & * & * \end{pmatrix} \end{aligned}$$

Now consider the following map determined by a symmetric matrix  $X$  and the action  $\beta$ :

$$\beta^\vee.X := \text{Upper}(n) \rightarrow \text{Sym}(n) : U \mapsto \sigma(UXU^{-1}).$$

Note that the image of this map is the orbit of  $X$  under the action  $\beta$ . We can differentiate the map  $\beta^\vee.X$  to get the following linear map:

$$D(\beta^\vee.X).I = \text{Tan}.\text{Upper}(n).I \rightarrow \text{Tan}.\text{Sym}(n).X : R \mapsto \sigma[R, X].$$

Note that the space tangent to the group  $\text{Upper}(n)$  at the identity may be identified with the space of upper triangular matrices; in symbols,

$$\text{Tan}.\text{Upper}(n).I = \text{upper}(n).$$

We can, and shall, regard the derivative  $D(\beta^\vee.X).I$  as a linear map from  $\text{upper}(n)$  to  $\text{Sym}(n)$ . It is not hard to prove that the space tangent to the orbit  $\beta.\text{Upper}(n).X$  at  $X$  is the image of this linear map; in symbols,

$$\text{Tan}.(\beta.\text{Upper}(n).X).X = \{\sigma[R, X] | R \in \text{upper}(n)\}.$$

Note that for smooth  $u : \text{Sym}(n) \rightarrow \text{upper}(n)$ , the vector field  $\tau.u$  is tangent to these orbits; in symbols, for all symmetric matrices  $X$ ,

$$\tau.u.X \in \text{Tan}.(\beta.\text{Upper}(n).X).X.$$

In fact, for symmetric  $X$ , the integral element  $\text{Lie.Staircase}.X$  of the system of vector fields  $\text{Lie.Staircase}$  at  $X$  is this tangent space; in symbols,

$$\text{Lie.Staircase}.X = \{\sigma[R, X] | R \in \text{upper}(n)\}.$$

In particular, note that, for any  $R \in \text{upper}(n)$ , we have the constant map  $u := \text{Sym}(n) \rightarrow \text{upper}(n) : Y \mapsto R$  and then  $\tau.u = \text{Sym}(n) \rightarrow \text{Sym}(n) : X \mapsto \sigma[X, R]$ . Thus the orbits of the action  $\beta$  of the upper triangular group are the integral submanifolds of the Lie algebra  $\text{Lie.Staircase}$  of vector fields. (In particular, note that the orbits are connected.)

### A Lie algebra of vector fields associated with inertia and congruence.

In this section I present an explicit description involving arbitrary square matrices, of a Lie algebra of vector fields on the space of symmetric matrices. By Nagano's submanifold theorem, this Lie algebra determines a partition of the space of symmetric matrices into integral submanifolds. I shall show that these submanifolds are orbits of the congruence action of the general linear group. The results in this section are a basis for analogous results in the next section.

Let  $j : \text{Sym}(n) \rightarrow R^{n \times n}$  be an analytic map. With  $j$ , we associate a vector field  $\gamma.j$  defined as follows:

$$\gamma.j := \text{Sym}(n) \rightarrow \text{Sym}(n) : X \mapsto X(j.X)^T + (j.X)X.$$

In particular, note that if  $X$  is symmetric and  $J$  is any square matrix with the same size as  $X$  then  $XJ^T + JX$  is symmetric.

**PROPOSITION.** *The set of all vector fields  $\gamma.j$ , such that  $j : \text{Sym}(n) \rightarrow R^{n \times n}$  is an analytic map, forms a Lie subalgebra of the Lie algebra of all analytic vector fields on the space of symmetric matrices. In particular, if  $j : \text{Sym}(n) \rightarrow R^{n \times n}$  and  $k : \text{Sym}(n) \rightarrow R^{n \times n}$  are analytic maps then  $[\gamma.j, \gamma.k] = \gamma.l$  where  $l : \text{Sym}(n) \rightarrow R^{n \times n}$  is defined by*

$$X \mapsto Dj.X.(\gamma.k.X) - Dk.X.(\gamma.j.X) + [j.X, k.X].$$

**PROOF.** Let  $L$  denote the set of vector fields defined in this proposition. Note that the map  $\gamma$  is linear. It follows that its image  $L$  is a vector space.

We now compute  $[\gamma.j, \gamma.k]$ . Note that for any pair  $X, Y$  of symmetric matrices, we have

$$D(\gamma.j).X.Y = Y(j.X)^T + (j.X)Y + X(Dj.X.Y)^T + (Dj.X.Y)X.$$

Using this formula, we have

$$\begin{aligned} [\gamma.j, \gamma.k].X &= D(\gamma.j).X.(\gamma.k.X) - D(\gamma.k).X.(\gamma.j.X) \\ &= (\gamma.k.X)(j.X)^T + (j.X)(\gamma.k.X) + X(Dj.X.(\gamma.k.X))^T + (Dj.X.(\gamma.k.X))X \\ &\quad - (\gamma.j.X)(k.X)^T - (k.X)(\gamma.j.X) - X(Dk.X.(\gamma.j.X))^T - (Dk.X.(\gamma.j.X))X \\ &= X(Dj.X.(\gamma.k.X) - Dk.X.(\gamma.j.X))^T + (Dj.X.(\gamma.k.X) - Dk.X.(\gamma.j.X))X \\ &\quad + X[j.X, k.X]^T + [j.X, k.X]X \end{aligned}$$

since

$$\begin{aligned} &(\gamma.k.X)(j.X)^T + (j.X)(\gamma.k.X) - (\gamma.j.X)(k.X)^T - (k.X)(\gamma.j.X) \\ &= (X(k.X)^T + (k.X)X)(j.X)^T + (j.X)(X(k.X)^T + (k.X)X) \\ &\quad - (X(j.X)^T + (j.X)X)(k.X)^T - (k.X)(X(j.X)^T + (j.X)X) \\ &= X[j.X, k.X]^T + [j.X, k.X]X. \end{aligned}$$

□

I shall use *Lie.Congrnce* to denote the Lie algebra of vector fields defined in this proposition; in symbols,

$$\text{Lie.Congruence} := \{\gamma.j | j : \text{Sym}(n) \rightarrow R^{n \times n} \text{ is analytic}\}.$$

I shall call this set the **Lie algebra of vector fields associated with congruence**. It follows from Nagano's submanifold theorem that this Lie algebra of vector fields determines a partition of the space of symmetric matrices into integral submanifolds. In fact these submanifolds are orbits of the congruence representation of the general linear group on the space of symmetric matrices as I shall now indicate.

Let  $Gl(n)$  denote the **general linear group** of invertible  $n$ -by- $n$  real matrices; in symbols,

$$Gl(n) := \{C \in R^{n \times n} | \det C \neq 0\}.$$

Consider the action of this group on the space of symmetric matrices defined by congruence:

$$\alpha := Gl(n) \times \text{Sym}(n) \rightarrow \text{Sym}(n) : (C, X) \mapsto CXC^T.$$

For a symmetric matrix  $X$  we have the orbit of  $X$  under this congruence action:

$$\alpha.Gl(n).X := \{CXCT : C \in Gl(n)\}.$$

Recall that two symmetric matrices  $X$  and  $Y$  are **congruent** if there is an invertible  $C$  such that  $Y = CXCT$ . Note that the orbit of a symmetric matrix  $X$  consists of the symmetric matrices which are congruent to  $X$ . For a symmetric matrix  $X$  the **inertia** of  $X$ , denoted by  $Inert.X$ , is the triple  $(np(X), nn(X), nz(X))$  where  $np(X)$  is the number of positive eigenvalues of  $X$ ,  $nn(X)$  is the number of negative eigenvalues of  $X$  and  $nz(X)$  is the number of zero eigenvalues of  $X$ . Recall that two symmetric matrices are congruent iff they have the same inertia. (See, for example, Birkhoff and MacLane [1953] Section IX.9 “Real quadratic forms under the full linear group”, or Horn and Johnson [1985] Section 4.5 “Congruence and simultaneous diagonalization of Hermitian and symmetric matrices”.) This result is usually called *Sylvester’s law of inertia*. Note that the diagonal matrices  $diag(1^{\times np}, (-1)^{\times nn}, 0^{\times nz})$  provide a canonical form for congruence.

Consider the following map determined by a symmetric matrix  $X$  and the representation  $\alpha$ :

$$\alpha^\vee.X := Gl(n) \rightarrow Sym(n) : C \mapsto CXCT.$$

Note that the image of this map is the orbit of  $X$  under the action  $\alpha$ . We can differentiate this map to get the following linear map:

$$D(\alpha^\vee.X).I = Tan.Gl(n).I \rightarrow Tan.Sym(n).X : K \mapsto KX + XK^T.$$

Recall that the space tangent to the group  $Gl(n)$  at the identity may be identified with the space on  $n$ -by- $n$  real matrices; in symbols,

$$Tan.Gl(n).I = R^{n \times n}.$$

(See, for example, Curtis [1984].) Also note that the space tangent to the space  $Sym(n)$  at  $X$  may be identified with the space  $Sym(n)$  itself; in symbols,

$$Tan.Sym(n).X = Sym(n).$$

Consequently, we can, and shall, regard the derivative  $D(\alpha^\vee.X).I$  as a linear map from  $R^{n \times n}$  to  $Sym(n)$ . It is not hard to prove that the the space tangent to the orbit at  $X$  is the image of this linear map. (Hint: For  $n$ -by- $n$  matrix  $K$  consider the curve  $t \mapsto exp(tK)$  in  $Gl(n)$ .) In symbols, we have

$$Tan.(\alpha.Gl(n).X).X = \{KX + XK^T | K \in R^{n \times n}\}.$$

Note that for smooth  $j : Sym(n) \rightarrow R^{n \times n}$  the vector field  $\gamma.j$  is tangent to these orbits; in symbols, for all symmetric matrices  $X$ ,

$$\gamma.j.X \in Tan.(\alpha.Gl(n).X).X.$$

In fact, for a symmetric matrix  $X$ , the integral element of the system of vector fields  $Lie.Congruence$  at  $X$  is this tangent space; in symbols,

$$Lie.Congruence.X = \{KX + XK^T | K \in R^{n \times n}\}.$$

In particular, note that, for any  $n$ -by- $n$  matrix  $K$ , we have the following constant map:

$$j := Sym(n) \rightarrow R^{n \times n} : Y \mapsto K$$

and hence

$$\gamma.j = \text{Sym}(n) \rightarrow \text{Sym}(n) : X \mapsto XK^T + KX.$$

Thus the orbits of the action  $\alpha$  are the integral manifolds of the Lie algebra *Lie.Congruence* of vector fields. (In particular, note that the orbits are connected.)

### A lie algebra of vector fields associated with displacement inertia and displaced congruence.

This section begins with a review of the definition of the displacement operator. Then I present an explicit description of a Lie algebras of vector fields (associated with “displaced congruence”) on the space of symmetric matrices which is closely related to the Lie algebra of such vector fields (associated with congruence) which appears in the last section. By Nagano’s submanifold theorem, this new Lie algebra determines a partition of the space of symmetric matrices into integral submanifolds. I shall show that these submanifolds are orbits of a representation of the general linear group which is equivalent to the congruence representation.

Let  $Z$  denote the  $n$ -by- $n$  **lower shift matrix** which is defined by  $Z(i, j) := \delta(i, j + 1)$ ; in other words,

$$Z := \begin{pmatrix} 0 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 1 & 0 \end{pmatrix}.$$

The **displacement operator**  $\mathcal{D}$  on the space of symmetric matrices is defined by

$$\mathcal{D} := \text{Sym}(n) \rightarrow \text{Sym}(n) : X \mapsto X - ZXZ^T.$$

It is not hard to prove that the map  $\mathcal{D}$  is invertible. (In particular, note that the map  $X \mapsto ZXZ^T$  is nilpotent.)

**REMARK.** Kailath, Kung and Morf [1979] introduced this displacement operator as a map on  $R^{n \times n}$ . They give an explicit formula for the inverse of the displacement operator in terms of outer products. For more on displacement structure see, for example, Heinig and Rost [1984], Kailath and Sayed [1999], Olshevsky [1997], and Olshevsky [2001].

Let  $j : \text{Sym}(n) \rightarrow R^{n \times n}$  be an analytic map. With  $j$ , we associate a vector field  $\gamma^{\mathcal{D}}.j$  defined as follows:

$$\begin{aligned} \gamma^{\mathcal{D}}.j &:= \text{Sym}(n) \rightarrow \text{Sym}(n) : \\ X &\mapsto \mathcal{D}^{-1}((\mathcal{D}.X)(j.(\mathcal{D}.X))^T + (j.(\mathcal{D}.X))(\mathcal{D}.X)). \end{aligned}$$

**PROPOSITION.** *The set of all vector fields  $\gamma^{\mathcal{D}}.j$ , such that  $j : \text{Sym}(n) \rightarrow R^{n \times n}$  is an analytic map, forms a Lie subalgebra of the Lie algebra of all analytic vector fields on the space of symmetric matrices. In particular, if  $j : \text{Sym}(n) \rightarrow R^{n \times n}$  and  $k : \text{Sym}(n) \rightarrow R^{n \times n}$  are analytic maps then  $[\gamma^{\mathcal{D}}.j, \gamma^{\mathcal{D}}.k] = \gamma^{\mathcal{D}}.l$  where  $l$  is defined as in the proposition in the last section.*

PROOF. This result follows from the corresponding result in the last section. I sketch the proof.

Note that  $\gamma^{\mathcal{D}}.j = \mathcal{D}^{-1} \circ (\gamma.j) \circ \mathcal{D}$ . Let  $u$  be any smooth vector field on  $Sym(n)$ . We calculate the following derivative:

$$D(\mathcal{D}^{-1} \circ u \circ \mathcal{D}).X.Y = \mathcal{D}^{-1}(Du.(\mathcal{D}.X).(\mathcal{D}.Y)).$$

Now let  $u$  and  $v$  be any two smooth vector fields on  $Sym(n)$ .

*Claim.*  $\mathcal{D}^{-1} \circ [u, v] \circ \mathcal{D} = [\mathcal{D}^{-1} \circ u \circ \mathcal{D}, \mathcal{D}^{-1} \circ v \circ \mathcal{D}]$ .

Recall the  $[u, v].Y = Du.Y.(v.Y) - Dv.Y.(u.Y)$ . Using the derivative formula calculated above we have

$$\begin{aligned} & [\mathcal{D}^{-1} \circ u \circ \mathcal{D}, \mathcal{D}^{-1} \circ v \circ \mathcal{D}].X \\ &= D(\mathcal{D}^{-1} \circ u \circ \mathcal{D}).X.((\mathcal{D}^{-1} \circ v \circ \mathcal{D}).X) \\ &\quad - D(\mathcal{D}^{-1} \circ v \circ \mathcal{D}).X.((\mathcal{D}^{-1} \circ u \circ \mathcal{D}).X) \\ &= \mathcal{D}^{-1}(Du.(\mathcal{D}.X).(v.(\mathcal{D}.X))) - \mathcal{D}^{-1}(Dv.(\mathcal{D}.X).(u.(\mathcal{D}.X))) \\ &= (\mathcal{D}^{-1} \circ [u, v] \circ \mathcal{D}).X. \end{aligned}$$

Finally then we have

$$\begin{aligned} [\gamma^{\mathcal{D}}.j, \gamma^{\mathcal{D}}.k] &= [\mathcal{D}^{-1} \circ (\gamma.j) \circ \mathcal{D}, \mathcal{D}^{-1} \circ (\gamma.k) \circ \mathcal{D}] \\ &= \mathcal{D}^{-1} \circ [\gamma.j, \gamma.k] \circ \mathcal{D} \\ &= \mathcal{D}^{-1} \circ (\gamma.l) \circ \mathcal{D} = \gamma^{\mathcal{D}}.l. \end{aligned}$$

□

I shall use  $Lie.Congruence^{\mathcal{D}}$  to denote the Lie algebra of vector fields defined in this proposition; in symbols,

$$Lie.Congruence^{\mathcal{D}} := \{\gamma^{\mathcal{D}}.j | j : Sym(n) \rightarrow R^{n \times n} \text{ is analytic}\}.$$

I call this set the **Lie algebra of vector fields associated with displaced congruence**. It follows from Nagano's submanifold theorem that this Lie algebra of vector fields determines a partition of the space  $Sym(n)$  into integral submanifolds. In fact these submanifolds are orbits of a representation of the general linear group on the space of symmetric matrices as I shall now indicate.

Recall that a **representation** of a group  $G$  on a vector space  $V$  is a homomorphism  $\rho : G \rightarrow Gl(V)$  of  $G$  into the group of invertible linear maps on  $V$ . A  **$G$ -linear map** of one representation  $\rho : G \rightarrow Gl(V)$  to a second representation  $\sigma : G \rightarrow Gl(W)$  is a linear map  $l : V \rightarrow W$  which satisfies  $\forall g \in G, l \circ (\rho.g) = (\sigma.g) \circ l$ ; in other words, we have a commutative square of linear maps for every element of  $G$ . (I have taken this terminology from Fulton and Harris [1991].) In this case, I write  $\sigma = l * \rho$ . Note that if  $l$  is invertible and  $G$ -linear then  $l^{-1}$  is  $G$ -linear. Two representations are **isomorphic** or **equivalent** if there is an invertible  $G$ -linear map between them.

**PROPOSITION.** *Let  $G$  be a group and let  $V$  and  $W$  be vector spaces. Let  $\sigma : G \rightarrow Gl(W)$  be a representation and let  $l : V \rightarrow W$  be an invertible linear map. Then the map*

$$\rho := G \rightarrow Gl(V) : g \mapsto l^{-1} \circ (\sigma.g) \circ l$$

*is a group representation which is equivalent to  $\sigma$ .*

PROOF. It is straight-forward to check that  $\rho$  is a representation. That  $\rho$  and  $\sigma$  are equivalent is obvious.  $\square$

If the hypotheses of this proposition are satisfied I say that such a representation  $\rho$  is **induced** by the representation  $\sigma$  and the invertible linear map  $l$ . Note that  $\sigma = l * \rho$  and  $\rho = l^{-1} * \sigma$ .

In the last section we discussed the congruence representation which is defined as follows:

$$\alpha := Gl(n) \rightarrow Gl(Sym(n)) : C \mapsto (X \mapsto CXC^T).$$

Since  $\mathcal{D} : Sym(n) \rightarrow Sym(n)$  is an invertible linear map, we have a representation of  $Gl(n)$  induced by  $\alpha$  and  $\mathcal{D}$ . I shall use  $\alpha^{\mathcal{D}}$  to denote this induced representation; in symbols,  $\alpha^{\mathcal{D}} = \mathcal{D} * \alpha$ . In particular, we have

$$\alpha^{\mathcal{D}} = Gl(n) \rightarrow Gl(Sym(n)) : C \mapsto (X \mapsto \mathcal{D}^{-1}(C(\mathcal{D}.X)C^T)).$$

I call  $\alpha^{\mathcal{D}}$  the **displaced congruence representation**.

Two symmetric matrices  $X$  and  $Y$  are **displacement-congruent** if  $\exists C \in Gl(n), Y = \mathcal{D}^{-1}(C(\mathcal{D}.X)C^T)$ . For a symmetric matrix  $X$ , the **displacement inertia** or **displaced inertia**,  $Inert^{\mathcal{D}}(X)$ , is defined to be the inertia of  $\mathcal{D}.X$ ; in symbols,  $Inert^{\mathcal{D}}(X) := Inert(\mathcal{D}.X)$ . Note that two symmetric matrices are displacement-congruent iff they have the same displacement inertia. Also note that the inverse displaced diagonal matrices  $\mathcal{D}^{-1}(diag(1^{n \times np}, (-1)^{n \times nn}, 0^{n \times nz}))$  provide a canonical form for displacement-congruence.

Consider the following map determined by symmetric matrix  $X$  and representation  $\alpha^{\mathcal{D}}$ :

$$\alpha^{\mathcal{D}\vee}.X := Gl(n) \rightarrow Sym(n) : C \mapsto \mathcal{D}^{-1}(C(\mathcal{D}.X)C^T).$$

Note that the image of this map is the orbit  $\alpha^{\mathcal{D}}.Gl(n).X$ . We can differentiate the map  $\alpha^{\mathcal{D}\vee}.X$  to get the following linear map:

$$\begin{aligned} D(\alpha^{\mathcal{D}\vee}.X).I &= Tan.Gl(n).I \rightarrow Tan.Sym(n).X : \\ K &\mapsto \mathcal{D}^{-1}.(K(\mathcal{D}.X) + (\mathcal{D}.X)K^T). \end{aligned}$$

We can, and shall, regard this derivative as a linear map from  $R^{n \times n} = Tan.Gl(n).I$  to  $Sym(n) = Tan.Sym(n).X$ .

It is not hard to prove that the space tangent to the orbit of  $X$  at  $X$  under the action of  $\alpha^{\mathcal{D}}$  is the image of this linear map. In symbols, we have

$$Tan.(\alpha^{\mathcal{D}}.Gl(n).X).X = \{\mathcal{D}^{-1}.(K(\mathcal{D}.X) + (\mathcal{D}.X)K^T) | K \in R^{n \times n}\}.$$

Note that for any smooth map  $j : Sym(n) \rightarrow R^{n \times n}$  the vector field  $\gamma^{\mathcal{D}}.j$  is tangent to these orbits; in symbols, for every symmetric matrix  $X$ ,

$$\gamma^{\mathcal{D}}.j.X \in Tan.(\alpha^{\mathcal{D}}.Gl(n).X).X.$$

In fact, for a symmetric matrix  $X$ , the integral element of the system of vector fields  $Lie.Congruence^{\mathcal{D}}$  at  $X$  is this tangent space; in symbols,

$$Lie.Congruence^{\mathcal{D}}.X = \{\mathcal{D}^{-1}.(K(\mathcal{D}.X) + (\mathcal{D}.X)K^T) | K \in R^{n \times n}\}.$$

Thus the orbits of the action  $\alpha^{\mathcal{D}}$  of the general linear group are the integral submanifolds of the Lie algebra  $Lie.Congruence^{\mathcal{D}}$  of vector fields.

**On symmetric matrices which have the same eigenvalues and the same staircase structure.**

In this section I present the first main result of this report. In previous sections, I have defined the Lie algebras *Lie.Spectrum* and *Lie.Staircase* of vector fields on the space of symmetric matrices associated respectively with eigenvalues and staircase structure. In this section we consider the intersection of these two Lie algebras. Since this intersection is a Lie algebra, by Nagano's submanifold theorem, the space of symmetric matrices is partitioned into a corresponding collection  $\mathcal{S}$  of integral submanifolds. This collection has the following property: For every symmetric matrix  $A$ , the submanifold of  $\mathcal{S}$  containing  $A$  consists of matrices which have the same eigenvalues as  $A$  and the same staircase structure as  $A$ .

Let  $M$  be a manifold and let  $L_1$  and  $L_2$  be subalgebras of the Lie algebra of all analytic vector fields on  $M$ . It is easy to see that the intersection  $L := L_1 \cap L_2$  is also a subalgebra. By Nagano's submanifold theorem this intersection determines a partition of the manifold  $M$  into integral submanifolds. It is also clear that, for any  $p$  in  $M$ , the integral submanifold determined by the intersection  $L$  and point  $p$  is a subset of the intersection of the integral submanifold determined by  $L_1$  and  $p$  and the integral submanifold determined by  $L_2$  and  $p$ ; in symbols,

$$\text{Subman}(L_1 \cap L_2, p) \subseteq \text{Subman}(L_1, p) \cap \text{Subman}(L_2, p).$$

If the intersection of the two submanifolds is not connected then the inclusion is proper. For example, the intersection  $L$  might be just the constant zero vector field; then the integral submanifolds of  $L$  are single points. Also note that the integral element determined by the intersection  $L$  and a point  $p$  is a subspace of the intersection of the integral element determined by  $L_1$  and  $p$  and the integral element determined by  $L_2$  and  $p$ ; in symbols,

$$(L_1 \cap L_2).p \subseteq (L_1.p) \cap (L_2.p).$$

Here too we might encounter proper inclusion between these vector spaces.

We now consider the Lie algebras mentioned at the beginning of this section. Recall that for  $j : \text{Sym}(n) \rightarrow \text{Skew}(n)$ ,  $\kappa.j$  denotes the vector field on  $\text{Sym}(n)$  defined by

$$\kappa.j := \text{Sym}(n) \rightarrow \text{Sym}(n) : X \mapsto [X, j.X]$$

and that the Lie algebra, *Lie.Spectra*, of vector fields on  $\text{Sym}(n)$  associated with spectra is the collection of all such  $\kappa.j$  where  $j$  is analytic. Also recall that for  $u : \text{Sym}(n) \rightarrow \text{upper}(n)$ ,  $\tau.u$  denotes the vector field on  $\text{Sym}(n)$  defined by

$$\tau.u := \text{Sym}(n) \rightarrow \text{Sym}(n) : X \mapsto \sigma[X, u.X]$$

where  $\sigma := R^{n \times n} \rightarrow \text{Sym}(n) : Y \mapsto Y_l + Y_d + Y_l^T$  and that the Lie algebra, *Lie.Staircase*, of vector fields associated with staircase structure is the collection of all such  $\tau.u$  where  $u$  is analytic. From above we know that *Lie.Spectrum* and *Lie.Staircase* are Lie subalgebras of the algebra of all analytic vector fields on  $\text{Sym}(n)$ . We now consider the intersection  $(\text{Lie.Spectra}) \cap (\text{Lie.Staircase})$  of these two subalgebras. We want to see that the intersection contains a substantial number of vector fields.

Let  $F := a_0 + a_1x + a_2x^2 + \dots$  be a (formal) power series with real coefficients. Recall that  $F$  has a radius of convergence,  $\rho(F)$ , which is given by the Cauchy-Hadamard formula:

$$\rho(F) = (\limsup_{k \rightarrow \infty} |a_k|^{1/k})^{-1}.$$

If  $\rho(F) = \infty$  then  $F$  is entire and we can use  $F$  to define the following analytic map on  $Sym(n)$ :

$$F := Sym(n) \rightarrow Sym(n) : X \mapsto F(X)$$

where  $F(X)$  is the sum of the infinite series obtained by substituting the symmetric matrix  $X$  for the indeterminate  $x$  in the (formal) power series. We shall also use  $F$  to denote this function on the space of symmetric matrices.

**PROPOSITION.** *The intersection  $(Lie.Spectrum) \cap (Lie.Staircase)$  of the Lie algebra of vector fields associated with spectra and the Lie algebra of vector fields associated with staircase structure is a Lie algebra of vector fields on the space of symmetric matrices. The space of symmetric matrices is partitioned into a corresponding collection  $S$  of connected integral submanifolds of this Lie algebra. This collection  $S$  has the following property: For every symmetric matrix  $A$ , the integral submanifold of  $S$  containing  $A$  consists of matrices which have the same eigenvalues as  $A$  and the same staircase structure as  $A$ . Furthermore, if  $F := a_0 + a_1x + a_2x^2 + \dots$  is a formal power series which is entire and  $j := Sym(n) \rightarrow Skew(n)$  is the analytic map defined by  $X \mapsto F(X)_l - F(X)_l^T$  then  $\kappa.j$  is an element of this intersection of Lie algebras.*

**PROOF.** It is easy to see that  $\kappa.j$  is in  $Lie.Spectra$ . We need to see that  $\kappa.j = \tau.u$  for some analytic map  $u : Sym(n) \rightarrow upper(n)$ .

Note that the space of real  $n$ -by- $n$  real matrices is the direct sum of the  $n$ -by- $n$  skew-symmetric matrices and the  $n$ -by- $n$  upper triangular matrices:  $R^{n \times n} = Skew(n) \oplus upper(n)$ . In particular, we have  $Y = (Y_l - Y_l^T) + (Y_d + Y_u + Y_l^T)$ . Using this decomposition, we have, for any symmetric matrix  $X$ ,

$$0 = [X, F(X)] = [X, F(X)_l - F(X)_l^T] + [X, F(X)_d + 2F(X)_l^T].$$

We can define  $u$  as follows:

$$u := Sym(n) \rightarrow upper(n) : X \mapsto -(F(X)_d + 2F(X)_l^T).$$

□

**REMARK.** Recall that the Toda flow is associated with the following differential equation:  $X' = [X_l - X_l^T, X]$ . Its vector field is among the ones mentioned in the proposition.

### On symmetric matrices which have the same eigenvalues and the same displacement inertia.

In this section, I present the second main result of this report. In previous sections, I have defined the Lie algebra  $Lie.Spectrum$  and the Lie algebra  $Lie.Congruence^D$  of vector fields on the space of symmetric matrices associated respectively with eigenvalues and displacement inertia. In this section, we consider the intersection of these two Lie algebras. Since this intersection is a Lie algebra, by Nagano's submanifold theorem, the space of symmetric matrices is

partitioned into a corresponding collection  $\mathcal{T}$  of connected integral submanifolds. Using the ideas of the last section we obtain the following result.

**PROPOSITION.** *The intersection  $(\text{Lie.Spectrum}) \cap (\text{Lie.Congruence}^{\mathcal{D}})$  of the Lie algebra of vector fields associated with spectra and the Lie algebra of vector fields associated with displacement inertia is a Lie algebra of vector fields on the space of symmetric matrices. The space of symmetric matrices is partitioned into a corresponding collection  $\mathcal{T}$  of connected integral submanifolds. This collection has the following property: For every symmetric matrix  $A$ , the submanifold of  $\mathcal{T}$  containing  $A$  consists of matrices which have the same eigenvalues as  $A$  and the same displacement inertia as  $A$ .*

I conjecture that the intersection described in this proposition contains a substantial number of vector fields.

### Acknowledgements.

I did most of this research during the Spring and Summer of 2001 while at the University of Wyoming. I wish to thank all the people in the Mathematics Department who made my affiliation possible, especially Benito Chen (the head of the department during the 2000-01 academic year) and Bryan Shader. I also wish to thank colleagues who discussed this research with me during that time and offered encouragement: Eric Moorhouse (University of Wyoming), Jaak Vilms (Colorado State University) and Peter Olver (University of Minnesota).

I presented a talk on this research at the 2001 AMS-IMS-SIAM Summer Research Conference on Fast Algorithms in Mathematics, Computer Science and Engineering at Mount Holyoke College in South Hadley, Massachusetts during August, 2001. I received some financial support from the conference sponsors which enabled me to attend this conference. I wish to thank the organizers of this conference, especially Vadim Olshevsky (University of Connecticut). I also wish to thank colleagues who attended this conference and discussed this research with me, especially Dario Fasino (Udine, Italy), Harry Dym (Weizmann Institute of Science, Rehovot, Israel), Miroslav Fiedler (Prague Academy of Science) and Tony Wen (MIT).

I also presented a talk on this research at the Fourth SIAM Conference on Linear Algebra in Signals, Systems and Control in Boston, Massachusetts in August, 2001. I wish to thank the organizers of this conference, especially Biswa Datta (Northern Illinois University). I also wish to thank colleagues who attended this conference and discussed this research with me, especially Peter Nylen (Auburn University), Uwe Helmke (University of Würzburg, Germany), Wasin So (San Jose State University) and Moody Chu (North Carolina State University).

### References

- Arbenz, P. and Golub, G. [1995], *Matrix shapes invariant under the symmetric QR algorithm*, Numerical Lin. Alg. with Applications **2**, 29-48.
- Ashlock, D.A.; Driessel, K.R. and Hentzel, I.R. [1997], *Matrix structures invariant under Toda-like isospectral flows*, Lin. Alg. and Applications **254**, 29-48.
- Birkhoff, G. and MacLane, S. [1953], *A Survey of Modern Algebra*, The Macmillan Company.
- Chu, M.T. [1984], *The generalized Toda flow, the QR algorithm and the center manifold theory*, SIAM J. Alg. Disc. Meth. **5**, 187-210.
- Chu, M.T. and Norris, L.K. [1988], *Isospectral flows and abstract matrix factorizations*, SIAM J. Numer. Anal. **25**, 1383-1391.
- Curtis, M.L. [1984], *Matrix Groups*, Springer.

- Davis, P.J. [1979], *Circulant Matrices*, John Wiley and Sons.
- Deift, P.; Nanda, T. and Tomei, C. [1983], *Ordinary differential equations and the symmetric eigenproblem*, SIAM J. Numer. Anal. **20**, 1-22.
- Demmel, J. [1997], *Applied Linear Algebra*, SIAM.
- Driessel, K.R. [1988], *On isospectral surfaces in the space of symmetric tridiagonal matrices*, Report No. URI-047, Department of Mathematical Sciences, Clemson University, Clemson, SC 29634, 28 pages.
- Driessel, K.R. [2000], *On computing canonical forms using flows*, preprint.
- Driessel, K.R. and Gerisch, A. [2001], *On some zero-preserving iso-spectral flows*, preprint, submitted to SIAM J. Matrix Analysis.
- Fasino, D. [2001], *Isospectral flows on displacement structured matrix spaces*, Structured Matrices: Recent Developments in Theory and Computation, edited by D. Bini, E. Tyrtyshnikov and P. Yalamov, Nova Science Publishers, Inc..
- Fulton, W. and Harris, J. [1991], *Representation Theory*, Springer.
- Hawkins, T. [2000], *Emergence of the Theory of Lie Groups*, Springer.
- Heinig, G. and Rost, K. [1984], *Algebraic Methods for Toeplitz-like Matrices and Operators*. Birkhäuser.
- Hermann, R. [1960], *On the differential geometry of foliations*. Annals of Math. **72**, 445-457.
- Hermann, R. [1962], *The differential geometry of foliations, II*, J. of Math. and Mech. **11**, 303-315.
- Horn, R.A. and Johnson, C.R. [1985], *Matrix Analysis*, Cambridge University Press.
- Isidori, A. [1985], *Nonlinear Control Systems: An Introduction*, Lecture Notes in Control and Information Sciences Number 72, Springer.
- Jurdjevic, V. [1997], *Geometric Control Theory*, Cambridge University Press.
- Kailath, T., Kung, S.-Y. and Morf, M. [1979], *Displacement ranks of a matrix*, AMS Bulletin **1**, 769-773.
- Kailath, T. and Sayed, A.H. (editors) [1999], *Fast Reliable Algorithms for Matrices with Structure*, SIAM.
- Nagano, T. [1966], *Linear differential systems with singularities and an application to transitive Lie algebras*, J. Math. Society Japan **18**, 398-404.
- Olshevsky, V. [1997], *Pivoting for structured matrices with applications*, preprint, to appear in Linear Algebra and Applications.
- Olshevsky, V. [2001] (editor), *Structured Matrices in Mathematics, Computer Science, and Engineering, Contemporary Mathematics, Volumes 280 and 281*, American Mathematical Society.
- Olver, P. [1995], *Equivalence, Invariants, and Symmetry*, Cambridge University Press.
- Symes, W. W. [1980a], *Systems of Toda type, inverse spectral problems, and representation theory*, Inventiones Math. **59**, 13-51.
- Symes, W. W. [1980b], *Hamiltonian group actions and integrable systems*, Physica **1D**, 339-374.
- Symes, W. W. [1982], *The QR algorithm and scattering for the finite nonperiodic Toda lattice*, Physica **4D**, 275-280.
- Watkins, D.S. [1984], *Isospectral flows*, SIAM Review **26**, 379-391.

MATHEMATICS DEPARTMENT, COLORADO STATE UNIVERSITY, FORT COLLINS, COLORADO.  
USA

*E-mail address:* driessel@math.colostate.edu

# Riccati Equations and Bitangential Interpolation Problems with Singular Pick Matrices

Harry Dym

**ABSTRACT.** A recently discovered connection between matrix Riccati equations and (finite dimensional) reproducing kernel Hilbert spaces is used to develop a clean and relatively painless analysis of a class of bitangential interpolation problems with singular Pick matrices.

## 1. Introduction

The objective of this paper is to exploit some recently discovered connections between Riccati equations and (finite dimensional) reproducing kernel Hilbert spaces in order to develop a clean and relatively painless analysis of bitangential interpolation problems with singular Pick matrices. The interpolation problems will be formulated in the Schur class  $\mathcal{S}^{p \times q}(\Omega)$  of  $p \times q$  mvf's (matrix valued functions)  $S(\lambda)$  that meet the following two conditions:

1.  $S(\lambda)$  is holomorphic (i.e., analytic) in  $\Omega$ .
2.  $S(\lambda)$  is contractive in  $\Omega$ , i.e.,

$$|\xi^* S(\lambda)\eta| \leq \|\xi\| \|\eta\|$$

for every point  $\lambda \in \Omega$  and every pair of vectors  $\xi \in \mathbb{C}^p$  and  $\nu \in \mathbb{C}^q$ .

For the sake of definiteness we shall choose  $\Omega = \mathbb{C}_+$ , the open upper half plane, and shall just write  $\mathcal{S}^{p \times q}$  instead of  $\mathcal{S}^{p \times q}(\Omega)$ . Much the same analysis goes through when  $\Omega$  is taken equal to the open right half plane or the open unit disc. A comprehensive account of bitangential interpolation problems in the disc that predated the Riccati equation approach, but still has many points of contact with this paper, is provided in [BD].

The problem of interest can be formulated most elegantly (and simply) in terms of three complex matrices  $C \in \mathbb{C}^{m \times n}$ ,  $A \in \mathbb{C}^{n \times n}$ ,  $P \in \mathbb{C}^{n \times n}$  which satisfy the following assumptions:<sup>1</sup>

---

1991 *Mathematics Subject Classification.* 46E22, 47A57, 30E05.

*Key words and phrases.* interpolation, Riccati equations, reproducing kernel Hilbert spaces, structured matrices, singular Pick matrices.

The author thanks Renee and Jay Weiss for endowing the chair that supports his research.

<sup>1</sup>We shall also assume that  $A$  is an upper triangular matrix in Jordan form. This involves no loss of generality, as will be explained shortly.

© 2003 American Mathematical Society

(A1)  $\sigma(A) \cap \mathbb{R} = \emptyset$ .

(A2)  $P$  is a positive semidefinite solution of the Lyapunov equation

$$(1.1) \quad A^*P - PA = 2\pi i C^*JC,$$

where

$$J = \begin{bmatrix} I_p & 0 \\ 0 & -I_q \end{bmatrix}, \quad p \geq 1, \quad q \geq 1, \quad p + q = m.$$

The assumption (A1) can be dropped, but then the problem becomes considerably more complicated.

The notation

$$(1.2) \quad F(\lambda) = C(\lambda I_n - A)^{-1}, \quad \lambda \in \mathbb{C} \setminus \sigma(A)$$

$$(1.3) \quad \Delta_S(\mu) = \begin{bmatrix} I_p & -S(\mu) \\ -S(\mu)^* & I_q \end{bmatrix} \text{ for a.e. } \mu \in \mathbb{R}$$

$$(1.4) \quad P_S = \int_{-\infty}^{\infty} F(\mu)^* \Delta_S(\mu) F(\mu) d\mu$$

will prove useful, as will the symbols mvf for matrix valued function,  $\sigma(A)$  for the spectrum of the matrix  $A$  and, for any mvf  $f(\lambda)$ ,

$f^\#(\lambda) = f(\bar{\lambda})^*$  and  $\mathfrak{H}_f$  for the set of points at which  $f$  is holomorphic.

Let  $\widehat{\mathcal{S}}(C, A, P)$  denote the set of mvf's  $S \in \mathcal{S}^{p \times q}$  that meet the following three conditions.

- (C1)  $[I_p \quad -S(\lambda)]F(\lambda)$  is holomorphic in the open upper half plane  $\mathbb{C}_+$ .
- (C2)  $[-S^\#(\lambda) \quad I_q]F(\lambda)$  is holomorphic in the open lower half plane  $\mathbb{C}_-$ .
- (C3)  $P_S \leq P$ .

In terms of this notation, our objective is to exhibit the role of Riccati equations in furnishing a complete description of the set  $\widehat{\mathcal{S}}(C, A, P)$  when  $P$  is singular.

It is readily seen that

$$\widehat{\mathcal{S}}(C, A, P) = \widehat{\mathcal{S}}(CT, T^{-1}AT, T^*PT)$$

for every invertible  $n \times n$  matrix  $T$ . (From a deeper point of view, this conclusion stems from the fact that it is the space

$$\mathcal{M} = \{F(\lambda)u : u \in \mathbb{C}^n\}$$

with “inner product”

$$\langle Fu, Fv \rangle_{\mathcal{M}} = v^*Pu$$

that is significant in the definition of the set  $\widehat{\mathcal{S}}(C, A, P)$ , and not the specific choice of  $T$  in the set  $\{CT, T^{-1}AT, T^*PT\}$ . Thus, without loss of generality, we can assume from the outset that  $A$  is in Jordan form. In fact, we can and shall assume that

$$A = \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix}$$

where the  $A_{jj}$ ,  $j = 1, 2$ , are in upper triangular Jordan form and  $\sigma(A_{11}) \subset \mathbb{C}_-$  and  $\sigma(A_{22}) \subset \mathbb{C}_+$ . Then the diagonal blocks in the corresponding block decomposition of  $P$  are uniquely determined by the Lyapunov equation (1.1). A couple of remarks are in order.

**REMARK 1.** *The particularly simple formulation of the conditions (C1) and (C2) is a consequence of the assumption that  $\sigma(A) \cap \mathbb{R} = \emptyset$ . If this restriction is dropped, then these conditions should be replaced by*

(C1a)  $[I_p \quad -S]Fu$  belongs to the Hardy space  $H_2^p$  for every  $u \in \mathbb{C}^n$ .

(C2a)  $[-S^\# \quad I_q]Fu$  is orthogonal to the Hardy space  $H_2^q$  for every  $u \in \mathbb{C}^n$ .

**REMARK 2.** *The assumption  $\sigma(A) \cap \mathbb{R} = \emptyset$  insures that if  $P$  is any solution of the Lyapunov equation (1.1), then*

$$P_S \leq P \iff P_S = P.$$

*A proof of this statement may be found e.g., in [D4].*

**REMARK 3.** *The problem of describing  $\hat{\mathcal{S}}(C, A, P)$  evolved from the author's study of the aBIP problem in [D4]. This problem can also be incorporated in the more general framework of the abstract interpolation problem of Katsnelson, Kheifets and Yuditskii [KKY]; see also [Kh] for additional references and extensions.*

The paper is organized as follows: In Section 2 we show that if  $S \in \mathbb{S}^{p \times q}$  meets the conditions (C1) and (C2), then  $P_S$  is a solution of the Lyapunov equation (1.1). In Section 3 a number of examples are discussed to illustrate the connection between conditions (C1) and (C2) and bitangential interpolation problems. In Section 4 we show that if, in addition to (A1)-(A2), it is also assumed that there exists a positive semi definite generalized inverse  $X$  of  $P$  that solves the Riccati equation (4.1), then the problem of interest splits naturally into two subproblems, each of which is then solved separately in Sections 6 and 7. In Section 8 we discuss the significance of the extra assumptions (that is labeled (A3)) and explain why the results are applicable even if it is not met. Section 5 surveys the background material that is needed to carry out the analysis in Sections 6-8.

The paper is largely self-contained. A strong attempt has been made to present the material under discussion in an elementary way. The only exception to this rule is Subsection 6.1, which can be skipped without losing the thread of the paper, if the reader is willing to accept the inclusion that is established in that subsection on faith. The final representation formula (7.10) for the set of solutions of the considered interpolation problem when the Pick matrix is singular is not new; see e.g., [BaH], [Du] and [BD] and for some special cases [D1]. However, the connection with Riccati equations that is exploited here does seem to be new. Moreover, this connection gives another explanation from the point of view of interpolation theory as to why Riccati equations play a significant role in  $H^\infty$  control and why they should play an important role in the study of (at least some classes of) structured matrices.

## 2. A preliminary problem

In order to add some perspective to the problem under consideration it is useful to first consider the following simpler question:

For what choices of  $C$  and  $A$  does there exist an  $S \in \mathcal{S}^{p \times q}$  that meets the conditions (C1) and (C2)?

A necessary condition rests on the following calculation:

**THEOREM 2.1.** *If assumption (A1) is in force and  $S \in \mathcal{S}^{p \times q}$  meets conditions (C1) and (C2), then the matrix  $P_S$  that is defined by formula (1.4) is a positive semidefinite solution of the Lyapunov equation (1.1)*

**PROOF.** The proof is purely computational and can be skipped without losing the main thread of the paper.

Let  $S \in \mathcal{S}^{p \times q}$  meet conditions (C1) and (C2) and for  $\varepsilon > 0$  let

$$(P_S)_\varepsilon = \int_{-\infty}^{\infty} F(\mu)^* \Delta_S(\mu) F(\mu) \frac{d\mu}{1 + i\varepsilon\mu}.$$

Then

$$\begin{aligned} & A^*(P_S)_\varepsilon - (P_S)_\varepsilon A \\ &= \int_{-\infty}^{\infty} \frac{\{(A^* - \mu I_n)F(\mu)^* \Delta_S(\mu)F(\mu) - F(\mu)^* \Delta_S(\mu)F(\mu)(A - \mu I_n)\}}{1 + i\varepsilon\mu} d\mu \\ &= 1_\varepsilon + 2_\varepsilon, \end{aligned}$$

where

$$1_\varepsilon = \frac{i}{\varepsilon} \int_{-\infty}^{\infty} C^* \Delta_S(\mu) F(\mu) \frac{d\mu}{\mu - i/\varepsilon}$$

and

$$2_\varepsilon = \frac{-i}{\varepsilon} \int_{-\infty}^{\infty} F(\mu)^* \Delta_S(\mu) C \frac{d\mu}{\mu - i/\varepsilon}.$$

The next step is to invoke the conditions (C1) and (C2) in order to evaluate  $1_\varepsilon$  and  $2_\varepsilon$  by Cauchy's theorem. Thus, upon writing

$$C = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix}$$

with  $C_1 \in \mathbb{C}^{p \times n}$  and  $C_2 \in \mathbb{C}^{q \times n}$ , we see that

$$\begin{aligned} 1_\varepsilon &= \lim_{R \uparrow \infty} \frac{i}{\varepsilon} \int_{-R}^R C_1^* [I_p - S(\mu)] F(\mu) \frac{d\mu}{\mu - i/\varepsilon} \\ &= 2\pi i(i/\varepsilon) C_1^* [I_p - S(i/\varepsilon)] F(i/\varepsilon) \\ &= 2\pi i(C_1^* C_1 - C_1^* S(i/\varepsilon) C_2)(I_n + i\varepsilon A)^{-1}, \end{aligned}$$

whereas

$$\begin{aligned} 2_\varepsilon &= \left\{ \lim_{R \uparrow \infty} \frac{i}{\varepsilon} \int_{-R}^R C^* \Delta_S(\mu) F(\mu) \frac{d\mu}{\mu + i/\varepsilon} \right\}^* \\ &= \left\{ \lim_{R \uparrow \infty} \frac{i}{\varepsilon} \int_{-R}^R C_2^* [-S^\#(\mu) - I_p] F(\mu) \frac{d\mu}{\mu + i/\varepsilon} \right\}^* \\ &= 2\pi i (I_n + i\varepsilon A^*)^{-1} (C_1^* S(i/\varepsilon) C_2 - C_2^* C_2). \end{aligned}$$

It is pretty clear that

$$\lim_{\varepsilon \downarrow 0} 2\pi i C_1^* C_1 (I_n + i\varepsilon A)^{-1} = 2\pi i C_1^* C_1$$

and

$$\lim_{\varepsilon \downarrow 0} -2\pi i (I_n + i\varepsilon A^*)^{-1} C_2^* C_2 = -2\pi i C_2^* C_2.$$

It is far from clear how to treat the terms involving  $S(i/\varepsilon)$  in  $1_\varepsilon$  and  $2_\varepsilon$  separately. The trick is to observe that the dominant contributions from  $1_\varepsilon$  and  $2_\varepsilon$  taken together cancel. To this end rewrite the terms involving  $S(i/\varepsilon)$  from  $1_\varepsilon$  and  $2_\varepsilon$  as

$$-2\pi i C_1^* S(i/\varepsilon) C_2 \{(I_n + i\varepsilon A)^{-1} - I_n\} + 2\pi i \{(I_n + i\varepsilon A^*)^{-1} - I_n\} C_1^* S(i/\varepsilon) C_2$$

and observe that it tends to zero as  $\varepsilon \downarrow 0$ , since  $S \in \mathcal{S}^{p \times q}$ .

We thus have shown that

$$\lim_{\varepsilon \downarrow 0} \{A^*(P_S)_\varepsilon - (P_S)_\varepsilon A\} = 2\pi i (C_1^* C_1 - C_2^* C_2).$$

This serves to complete the proof, since

$$\lim_{\varepsilon \downarrow 0} (P_S)_\varepsilon = P_S.$$

□

**THEOREM 2.2.** *If assumption (A1) is in force and  $S \in \mathcal{S}^{p \times q}$  meets the conditions (C1) and (C2), then the Lyapunov equation (1.1) admits at least one positive semidefinite solution  $P \geq P_S$ .*

**PROOF.** This is an immediate consequence of Theorem 2.1. □

### 3. Examples

In this section we will present a few examples to illustrate the connection between conditions (C1) and (C2) and bitangential interpolation problems.

**EXAMPLE 1.** Let

$$C = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} = \begin{bmatrix} \xi_1 & \cdots & \xi_n \\ \eta_1 & \cdots & \eta_n \end{bmatrix} \text{ and } A = \text{diag}\{\overline{\alpha_1}, \dots, \overline{\alpha_n}\},$$

where

$$\xi_j \in \mathbb{C}^p \text{ and } \eta_j \in \mathbb{C}^q \text{ for } j = 1, \dots, n,$$

$$\alpha_j \in \mathbb{C}_+ \text{ for } j = 1, \dots, \ell \text{ and } \alpha_j \in \mathbb{C}_- \text{ for } j = \ell + 1, \dots, n.$$

Then a mvf  $S \in \mathcal{S}^{p \times q}$  satisfies condition (C1) if and only if

$$\frac{\xi_j - S(\lambda)\eta_j}{\lambda - \bar{\alpha}_j} \text{ is holomorphic in } \mathbb{C}_+ \text{ for } j = 1, \dots, n.$$

For  $j = 1, \dots, \ell$ , this is automatically so. For  $j = \ell + 1, \dots, n$ , condition (C1) is met if and only if

$$\xi_j = S(\bar{\alpha}_j)\eta_j \text{ for } j = \ell + 1, \dots, n.$$

Similarly,  $S(\lambda)$  will meet condition (C2) if and only if

$$\frac{-S^\#(\lambda)\xi_j + \eta_j}{\lambda - \bar{\alpha}_j} \text{ is holomorphic in } \mathbb{C}_- \text{ for } j = 1, \dots, n,$$

or, equivalently, if and only if

$$\frac{\xi_j^* S(\lambda) - \eta_j^*}{\lambda - \alpha_j} \text{ is holomorphic in } \mathbb{C}_+ \text{ for } j = 1, \dots, n.$$

But this is so if and only if

$$\xi_j^* S(\alpha_j) = \eta_j^* \text{ for } j = 1, \dots, \ell.$$

EXAMPLE 2. Let

$$C = \begin{bmatrix} \xi_1 & \cdots & \xi_n \\ \eta_n & \cdots & \eta_n \end{bmatrix} \text{ and } A = \begin{bmatrix} \bar{\alpha} & 1 & 0 & 0 & \cdots & 0 \\ 0 & \bar{\alpha} & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & 0 & 0 & \cdots & \bar{\alpha} \end{bmatrix},$$

where  $A$  is an  $n \times n$  matrix with  $\bar{\alpha}$ 's on the main diagonal, 1's on the first superdiagonal and 0 elsewhere. Then

$$F(\lambda) = C(\lambda I_n - A)^{-1} = \begin{bmatrix} g_1(\lambda) & \cdots & g_n(\lambda) \\ h_1(\lambda) & \cdots & h_n(\lambda) \end{bmatrix},$$

where

$$g_k(\lambda) = \sum_{j=1}^k \frac{\xi_j}{(\lambda - \bar{\alpha})^{k+1-j}} \text{ and } h_k(\lambda) = \sum_{j=1}^k \frac{\eta_j}{(\lambda - \bar{\alpha})^{k+1-j}}$$

for  $k = 1, \dots, n$ .

Suppose first that  $\alpha \in \mathbb{C}_+$ . Then  $S \in \mathcal{S}^{p \times q}$  automatically meets condition (C1) and will meet (C2) if and only if

$$g_k^\#(\lambda)S(\lambda) - h_k^\#(\lambda) \text{ is holomorphic in } \mathbb{C}_+$$

for  $k = 1, \dots, n$ , i.e., if only if

$$\left\{ \sum_{j=1}^k \frac{\xi_j^*}{(\lambda - \alpha)^{j+k-1}} \right\} S(\lambda) - \sum_{j=1}^k \frac{\eta_j^*}{(\lambda - \alpha)^{j+k-1}}$$

is holomorphic in  $\mathbb{C}_+$  for  $k = 1, \dots, n$ . But this will be the case if and only if

$$\begin{aligned}\xi_1^* S(\alpha) &= \eta_1^* \\ \xi_1^* S^{(1)}(\alpha) + \xi_2^* S(\alpha) &= \eta_2^* \\ &\vdots \\ \xi_1^* \frac{S^{(n-1)}(\alpha)}{(n-1)!} + \cdots + \xi_n^* S(\alpha) &= \eta_n^*.\end{aligned}$$

This can be written neatly in terms of the upper triangular block Toeplitz matrix

$$\Lambda_\alpha = \begin{bmatrix} a_1 & a_2 & \cdots & a_n \\ 0 & a_1 & \cdots & a_{n-1} \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & a_1 \end{bmatrix}$$

with

$$a_j = \frac{S^{(j-1)}(\alpha)}{(j-1)!} \text{ for } j = 1, \dots, n.$$

Then the preceding condition is just

$$[\xi_1^* \cdots \xi_n^*] \Lambda_\alpha = [\eta_1^* \cdots \eta_n^*].$$

Similar considerations lead to the supplementary conclusion that if  $\alpha \in \mathbb{C}_-$ , then  $S \in \mathcal{S}^{p \times q}$  automatically meets (C2) but will meet (C1) if and only if

$$\begin{bmatrix} b_1 & 0 & \cdots & 0 \\ b_2 & b_1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ b_n & b_{n-1} & \cdots & b_1 \end{bmatrix} \begin{bmatrix} \eta_1 \\ \vdots \\ \eta_n \end{bmatrix} = \begin{bmatrix} \xi_1 \\ \vdots \\ \xi_n \end{bmatrix},$$

where

$$b_j = \frac{S^{(j-1)}(\bar{\alpha})}{(j-1)!} \text{ for } j = 1, \dots, n.$$

EXAMPLE 3. Let

$$C = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} = \begin{bmatrix} \xi_1 & \xi_2 & \xi_3 \\ \eta_1 & \eta_2 & \eta_3 \end{bmatrix} \text{ and } A = \begin{bmatrix} \bar{\alpha} & 1 & 0 \\ 0 & \bar{\alpha} & 0 \\ 0 & 0 & \alpha \end{bmatrix},$$

where  $\xi_j \in \mathbb{C}^p$  and  $\eta_j \in \mathbb{C}^q$  for  $j = 1, \dots, 3$  and  $\alpha \in \mathbb{C}_+$ . Then

$$(3.1) \quad F(\lambda) = C(\lambda I_n - A)^{-1} = \begin{bmatrix} g_1(\lambda) & g_2(\lambda) & g_3(\lambda) \\ h_1(\lambda) & h_2(\lambda) & h_3(\lambda) \end{bmatrix},$$

where

$$(3.2) \quad g_1(\lambda) = \frac{\xi_1}{\lambda - \bar{\alpha}}, \quad g_2(\lambda) = \frac{\xi_1}{(\lambda - \bar{\alpha})^2} + \frac{\xi_2}{\lambda - \bar{\alpha}}, \quad g_3(\lambda) = \frac{\xi_3}{\lambda - \alpha}$$

and

$$(3.3) \quad h_1(\lambda) = \frac{\eta_1}{\lambda - \bar{\alpha}}, \quad h_2(\lambda) = \frac{\eta_1}{(\lambda - \bar{\alpha})^2} + \frac{\eta_2}{\lambda - \bar{\alpha}}, \quad h_3(\lambda) = \frac{\eta_3}{\lambda - \alpha}.$$

Thus, as  $g_j(\lambda)$  and  $h_j(\lambda)$  are holomorphic in  $\mathbb{C}_+$  for  $j = 1, 2$  and  $g_3(\lambda)$  and  $h_3(\lambda)$  are holomorphic in  $\mathbb{C}_-$ , it is readily checked that if  $S \in \mathcal{S}^{p \times q}$ , then:

$$\begin{aligned} S \text{ meets condition (C1)} &\iff g_3(\lambda) - S(\lambda)h_3(\lambda) \text{ is holomorphic in } \mathbb{C}_+ \\ &\iff S(\alpha)\eta_3 = \xi_3 \end{aligned}$$

and

$$\begin{aligned} S \text{ meets condition (C2)} &\iff -S^\#(\lambda)g_j(\lambda) + h_j(\lambda) \\ &\quad \text{is holomorphic in } \mathbb{C}_- \text{ for } j = 1, 2 \\ &\iff g_j^\#(\lambda)S(\lambda) - h_j^\#(\lambda) \\ &\quad \text{is holomorphic in } \mathbb{C}_+ \text{ for } j = 1, 2 \\ &\iff \xi_1^*S(\alpha) = \eta_1^* \text{ and } \xi_1^*S'(\alpha) + \xi_2^*S(\alpha) = \eta_2^*. \end{aligned}$$

If  $\sigma(A) \cap \sigma(A^*) \neq \emptyset$ , as in this example, then the Lyapunov equation (1.1) does not have a unique solution. Indeed, it is readily checked that if  $P = [p_{ij}]$ ,  $i, j = 1, \dots, 3$ , is a Hermitian solution of equation (1.1), then all the entries  $p_{ij}$  of  $P$  are uniquely determined by the data  $C$  and  $A$ , except for  $p_{23}$  (and  $p_{32} = \overline{p_{23}}$ ). (This is not an accident. It stems from the fact that the diagonal blocks  $A_{11}$  and  $A_{22}$  of  $A$  meet the condition  $\sigma(A_{11}) \cap \sigma(A_{11}^*) = \emptyset$  and  $\sigma(A_{22}) \cap \sigma(A_{22}^*) = \emptyset$  and hence the corresponding block entries in  $P$  are uniquely determined; [Bh] is a good source of information on Lyapunov equations and Sylvester equations.) In the case at hand, an easy computation reveals that  $p_{13}$  is also uniquely determined but that  $p_{23}$  is “free”, and is subject only to the constraint that the full matrix  $P$  should be positive semidefinite. This permits another interpolation condition to be imposed via (C3): In terms of the standard basis  $e_1, e_2, e_3$  of  $\mathbb{C}^3$  and formulas (3.2) and (3.3) for the components of  $F(\lambda)$ ,

$$p_{ij} = e_i^* P_S e_j = \int_{-\infty}^{\infty} [g_i(\mu)^* \quad h_i(\mu)^*] \Delta_S(\mu) \begin{bmatrix} g_j(\mu) \\ h_j(\mu) \end{bmatrix} d\mu.$$

Consequently,

$$\begin{aligned} p_{13} &= \int_{-\infty}^{\infty} [\xi_1^* \quad \eta_1^*] \frac{\Delta_S(\mu)}{(\mu - \alpha)^2} \begin{bmatrix} \xi_3 \\ \eta_3 \end{bmatrix} d\mu \\ &= \int_{-\infty}^{\infty} \{ \xi_1^*(\xi_3 - S(\mu)\eta_3) + \eta_1^*(-S(\mu)^*\xi_3 + \eta_3) \} \frac{d\mu}{(\mu - \alpha)^2} \\ &= -2\pi i \xi_1^* S'(\alpha) \eta_3 \end{aligned}$$

and

$$\begin{aligned} p_{23} &= \int_{-\infty}^{\infty} \left\{ \frac{[\xi_1^* - \eta_1^*]}{(\mu - \alpha)^2} + \frac{[\xi_2^* - \eta_2^*]}{(\mu - \alpha)} \right\} \begin{bmatrix} \xi_3 - S(\mu)\eta_3 \\ -S(\mu)^*\xi_3 + \eta_3 \end{bmatrix} \frac{d\mu}{\mu - \alpha} \\ &= \int_{-\infty}^{\infty} \xi_1^*(\xi_3 - S(\mu)\eta_3) \frac{d\mu}{(\mu - \alpha)^3} - 2\pi i \xi_2^* S'(\alpha) \eta_3 \\ &= -\pi i \xi_1^* S''(\alpha) \eta_3 - 2\pi i \xi_2^* S'(\alpha) \eta_3. \end{aligned}$$

#### 4. Riccati Equations

Our next objective is to obtain a description of the set  $\widehat{\mathcal{S}}(C, A, P)$  under the following extra assumption:

(A3) There exists an  $n \times n$  complex matrix  $X$  such that:

1.  $X \geq 0$ .
2.  $X$  solves the Riccati equation

$$(4.1) \quad XA^* - AX = 2\pi i XC^* JCX.$$

3.  $XPX = X$ .
4.  $PXP = P$ .

If  $P$  is invertible, then the choice  $X = P^{-1}$  fulfills all four requirements. If  $P$  is not invertible, then there is no guarantee that such an  $X$  exists. If it does, then in view of (1), (3) and (4), it must be a positive semidefinite pseudoinverse of  $P$  that has the same rank as  $P$ . But that is not enough,  $\mathcal{R}_X$ , the range of  $X$ , must be invariant under  $A$ :

LEMMA 4.1. *If (A2) is in force and  $X$  is any  $n \times n$  Hermitian matrix such that  $XPX = X$ , then the following conditions are equivalent:*

1.  $AX = XPAX$ .
2.  $A\mathcal{R}_X \subseteq \mathcal{R}_X$ , i.e.,  $AX = XA_0$  for some matrix  $A_0$ .
3.  $X$  is a solution of the Riccati equation (4.1).
4.  $AX = XZAX$  for any matrix  $Z$  such that  $XZX = X$ .

PROOF. The implications (1)  $\Rightarrow$  (2), (4)  $\Rightarrow$  (2) and (3)  $\Rightarrow$  (1) are selfevident. Suppose next that (2) is in force and  $XZX = X$ . Then

$$XZAX = XZX A_0 = XA_0 = AX.$$

Therefore, (2)  $\Rightarrow$  (4) and (2)  $\Rightarrow$  (1).

Finally, upon multiplying both sides of the Lyapunov equation (1.1) by  $X$ , we obtain

$$XA^*PX - XPAX = 2\pi i XC^* JCX.$$

But this leads easily to the conclusion that (1)  $\Rightarrow$  (3). □

For any  $n \times n$  matrix  $E$ , let

$$\mathcal{N}_E = \{u \in \mathbb{C}^n : Eu = 0\} \text{ and } \mathcal{R}_E = \{Eu : u \in \mathbb{C}^n\}.$$

LEMMA 4.2. *If assumptions (A1)-(A3) are in force<sup>2</sup>, then*

$$(4.2) \quad \mathbb{C}^n = \mathcal{N}_P + \mathcal{R}_X .$$

PROOF. Let  $u \in \mathcal{N}_P \cap \mathcal{R}_X$ . Then  $Pu = 0$  and  $u = Xy$  for some  $y \in \mathbb{C}^n$ . Therefore,

$$u = Xy = XPXy = XPu = 0 .$$

This proves that  $\mathcal{N}_P \cap \mathcal{R}_X = \{0\}$ , i.e., the sum is direct. To complete the proof, it suffices to observe that

$$n = \dim \mathcal{N}_P + \dim \mathcal{R}_P = \dim \mathcal{N}_P + \dim \mathcal{R}_X ,$$

since  $\text{rank } P = \text{rank } X$ .  $\square$

From now on we shall let  $X^\dagger$  denote the Moore-Penrose pseudo inverse of  $X$ . Thus, if  $\text{rank } X = k$  and

$$X = U \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} U^* = [U_1 \quad U_2] \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_1^* \\ U_2^* \end{bmatrix} = U_1 D U_1^* ,$$

where  $U$  is unitary and  $D$  is a positive definite  $k \times k$  matrix, then

$$X^\dagger = U \begin{bmatrix} D^{-1} & 0 \\ 0 & 0 \end{bmatrix} U^* = U_1 D^{-1} U_1^* .$$

In view of Lemma 4.2, we see that if (A1)-(A3) are in force, then

$$(4.3) \quad AX = XX^\dagger AX \text{ and } XA^* = XA^*X^\dagger X .$$

In fact, if only (A2) and (3) of (A3) are in force, then

$X$  meets the conditions (4.3)  $\iff X$  is a solution of equation (4.1).

The decomposition (4.2) of  $\mathbb{C}^n$  enables us to split the study of  $\widehat{\mathcal{S}}(C, A, P)$  into two subproblems. To explain the first, let  $\widehat{\mathcal{S}}_X(C, A, P)$  denote the set of  $S \in \mathcal{S}^{p \times q}$  that meet the following three conditions:

- (C4)  $[I_p \quad -S(\lambda)]F(\lambda)X$  is analytic in  $\mathbb{C}_+$ .
- (C5)  $[-S^\#(\lambda) \quad I_q]F(\lambda)X$  is analytic in  $\mathbb{C}_-$ .
- (C6)  $X P_S X \leq X$ .

In view of the constraint  $X P_X X = X$ , it is clear that

$$\widehat{\mathcal{S}}(C, A, P) \subseteq \widehat{\mathcal{S}}_X(C, A, P) .$$

Next, let  $\widehat{\mathcal{S}}_N(C, A, P)$  denote the set of  $S \in \mathcal{S}^{p \times q}$  that meet the following two conditions:

- (C7)  $[I_p \quad -S(\lambda)]F(\lambda)v \equiv 0$  in  $\mathbb{C}_+$  for every  $v \in \mathcal{N}_P$ .
- (C8)  $[-S^\#(\lambda) \quad I_q]F(\lambda)v \equiv 0$  in  $\mathbb{C}_-$  for every  $v \in \mathcal{N}_P$ .

---

<sup>2</sup>The lemma is formulated this way in order to ease the exposition. However, the conclusion is valid for any pair of  $n \times n$  matrices  $P$  and  $X$  of the same rank for which  $X P_X X = X$ .

If  $S \in \widehat{\mathcal{S}}(C, A, P)$  and  $v \in \mathcal{N}_P$ , then

$$v^* P_S v \leq v^* P v = 0.$$

Therefore, since  $\Delta_S(\mu) \geq 0$  a.e. on  $\mathbb{R}$ , it follows immediately from the definition of  $P_S$  that  $\Delta_S(\mu)F(\mu)v = 0$  for a.e. point  $\mu \in \mathbb{R}$  and hence that

$$\widehat{\mathcal{S}}(C, A, P) \subseteq \widehat{\mathcal{S}}_N(C, A, P).$$

**THEOREM 4.1.** *If (A1)-(A3) are in force, then*

$$\widehat{\mathcal{S}}(C, A, P) = \widehat{\mathcal{S}}_X(C, A, P) \cap \widehat{\mathcal{S}}_N(C, A, P).$$

**PROOF.** Let  $S(\lambda)$  belong to the intersection of the two sets on the right and let  $u \in \mathbb{C}^n$ . Then, in view of Lemma 4.2,

$$u = Xy + v$$

for some  $y \in \mathbb{C}^n$  and some  $v \in \mathcal{N}_P$ . Therefore,

$$F(\lambda)u = F(\lambda)Xy + F(\lambda)v$$

and hence  $S(\lambda)$  clearly meets the conditions (C1) and (C2). Moreover, since

$$u^* P_S u = y^* X P_S X y \leq y^* X y = u^* P u,$$

we see that

$$\widehat{\mathcal{S}}_X(C, A, P) \cap \widehat{\mathcal{S}}_N(C, A, P) \subseteq \widehat{\mathcal{S}}(C, A, P).$$

Therefore, since the opposite inclusion was established in the discussion preceding the statement of the theorem, the proof is complete.  $\square$

## 5. Some background material

In this section, we present a brief outline of some of the background material that will be needed for the subsequent analysis. Proofs and additional information on the general facts that are presented in the first two subsections may be found in many sources; see e.g., [BGR] and [D1]. The next two subsections deal with formulas that are based on solutions  $X$  to the Riccati equation (4.1) that are developed in [D5] and [D6]. The last subsection recalls the definition of a reproducing kernel Hilbert space, for use in Subsection 6.1 (though in fact the theory of such spaces underlies many of the calculations that are carried out in this paper).

**5.1. Rational  $J$ -inner mvf's.** A rational  $m \times m$  mvf  $\Theta(\lambda)$  is said to be  $J$ -inner (with respect to  $\mathbb{C}_+$ ) if

$$\Theta(\lambda)^* J \Theta(\lambda) \leq J \text{ for } \lambda \in \mathfrak{H}_\Theta \cap \mathbb{C}_+$$

and

$$\Theta(\mu)^* J \Theta(\mu) = J \text{ for } \mu \in \mathfrak{H}_\Theta \cap \mathbb{R}.$$

If  $\Theta(\lambda)$  is written in block form as

$$\Theta(\lambda) = \begin{bmatrix} \Theta_{11}(\lambda) & \Theta_{12}(\lambda) \\ \Theta_{21}(\lambda) & \Theta_{22}(\lambda) \end{bmatrix}$$

with diagonal blocks  $\Theta_{11}(\lambda)$  of size  $p \times p$  and  $\Theta_{22}(\lambda)$  of size  $q \times q$ , then  $\Theta_{22}(\lambda)$  is invertible at every point  $\lambda \in \mathfrak{H}_\Theta \cap \overline{\mathbb{C}_+}$  and (the so-called Potapov-Ginzburg transform)

$$\begin{aligned}\Sigma(\lambda) &= \begin{bmatrix} \Theta_{11}(\lambda) & \Theta_{12}(\lambda) \\ 0 & I_p \end{bmatrix} \begin{bmatrix} I_q & 0 \\ \Theta_{21}(\lambda) & \Theta_{22}(\lambda) \end{bmatrix}^{-1} \\ &= \begin{bmatrix} \Sigma_{11}(\lambda) & \Sigma_{12}(\lambda) \\ \Sigma_{21}(\lambda) & \Sigma_{22}(\lambda) \end{bmatrix}\end{aligned}$$

is a rational  $m \times m$  inner mvf in  $\mathbb{C}_+$ , i.e.,

$$\Sigma(\lambda)^* \Sigma(\lambda) \leq I_m \text{ for } \lambda \in \mathbb{C}_+,$$

with equality on the boundary. In particular  $\Sigma_{11} \in \mathcal{S}^{p \times p}$ ,  $\Sigma_{12} \in \mathcal{S}^{p \times q}$ ,  $\Sigma_{21} \in \mathcal{S}^{q \times p}$  and  $\Sigma_{22} \in \mathcal{S}^{q \times q}$ .

**5.2. Linear fractional transformations.** If  $\Theta(\lambda)$  is  $J$ -inner then the linear fractional transformation

$$(5.1) \quad T_\Theta[\varepsilon] = (\Theta_{11}\varepsilon + \Theta_{12})(\Theta_{21}\varepsilon + \Theta_{22})^{-1}$$

maps  $\varepsilon \in \mathcal{S}^{p \times q}$  into  $\mathcal{S}^{p \times q}$ .

There are two additional formulas for  $T_\Theta[\varepsilon]$  that are useful:

$$(5.2) \quad T_\Theta[\varepsilon] = ((\Theta_{11})^\# + \varepsilon(\Theta_{12})^\#)^{-1}((\Theta_{21})^\# + \varepsilon(\Theta_{22})^\#)$$

and

$$(5.3) \quad T_\Theta[\varepsilon] = \Sigma_{12} + \Sigma_{11}\varepsilon(I_q - \Sigma_{21}\varepsilon)^{-1}\Sigma_{22}.$$

The formula (5.2) is obtained easily from (5.1) by noting that

$$[I_p \quad \varepsilon(\lambda)]\Theta^\#(\lambda)J\Theta(\lambda) \begin{bmatrix} \varepsilon(\lambda) \\ I_q \end{bmatrix} = [I_p \quad \varepsilon(\lambda)]J \begin{bmatrix} \varepsilon(\lambda) \\ I_q \end{bmatrix} = 0$$

for every point  $\lambda \in \mathbb{C}_+$  at which  $\Theta(\lambda)$  and  $\Theta^\#(\lambda)$  are both holomorphic and then writing out the left hand side in terms of the blocks of  $\Theta(\lambda)$ . Formula (5.3) is often referred to as the Redheffer transform and denoted by  $R_\Sigma[\varepsilon]$ . It can be obtained by reexpressing formula (5.1) in terms of the blocks  $\Sigma_{ij}(\lambda)$ .

**5.3. Specific formulas.** If (A1)-(A3) are in force, then the mvf

$$(5.4) \quad \Theta_X(\lambda) = I_m - 2\pi i C(\lambda I_n - A)^{-1} X C^* J$$

is  $J$ -inner and the corresponding Potapov-Ginzburg transform

$$(5.5) \quad \Sigma_X(\lambda) = I_m - 2\pi i J C(\lambda I_n - A_2)^{-1} X C^* J,$$

where

$$A_2 = A - 2\pi i X C_2^* C_2.$$

**5.4. Minimal realizations.** It is not hard to show that the realization formulas (5.4) and (5.5) are minimal if and only if  $X$  is invertible. Our next objective is to obtain realizations that are minimal even when  $X$  is not invertible by taking advantage of the invariance condition

$$AX = XX^\dagger AX$$

and the decomposition

$$(5.6) \quad X = U \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} U^* = U_1 D U_1^*,$$

in which  $D$  is a  $k \times k$  positive definite matrix and  $U_1^* U_1 = I_k$ . The following notation will be useful:

$$(5.7) \quad \tilde{A} = U_1^* A U_1, \quad \tilde{C} = C U_1, \quad \tilde{C}_1 = C_1 U_1, \quad \tilde{C}_2 = C_2 U_1,$$

$$(5.8) \quad A_1 = A + 2\pi i X C_1^* C_1, \quad A_2 = A - 2\pi i X C_2^* C_2,$$

$$(5.9) \quad \tilde{A}_1 = U_1^* A_1 U_1 = \tilde{A} + 2\pi i D \tilde{C}_1^* \tilde{C}_1$$

and

$$(5.10) \quad \tilde{A}_2 = U_1^* A_2 U_1 = \tilde{A} - 2\pi i D \tilde{C}_2^* \tilde{C}_2.$$

LEMMA 5.1. *If (A1)-(A3) are in force, then:*

1.  $X U_1 = U_1 (U_1^* X U_1) = U_1 D$ .
2.  $A U_1 = U_1 \tilde{A}$ .
3.  $\sigma(\tilde{A}) \subset \sigma(A)$ .
4.  $(\lambda I_n - A)^{-1} U_1 = U_1 (\lambda I_k - \tilde{A})^{-1}$  if  $\lambda \notin \sigma(A)$ .
5.  $A_j U_1 = U_1 \tilde{A}_j$ ,  $j = 1, 2$ .
6.  $\sigma(\tilde{A}_j) \subset \sigma(A_j)$ ,  $j = 1, 2$ .
7.  $(\lambda I_n - A_j)^{-1} U_1 = U_1 (\lambda I_k - \tilde{A}_j)^{-1}$  if  $\lambda \notin \sigma(A_j)$ ,  $j = 1, 2$ .
8.  $\tilde{A}_1^* = D^{-1} \tilde{A}_2 D$ .
- 9.

$$(5.11) \quad \tilde{A}^* D^{-1} - D^{-1} \tilde{A} = 2\pi i \tilde{C}^* J \tilde{C}.$$

$$10. \quad \tilde{A}^* D^{-1} - D^{-1} \tilde{A}_1 = -2\pi i \tilde{C}_2^* \tilde{C}_2.$$

$$11. \quad \tilde{A}^* D^{-1} - D^{-1} \tilde{A}_2 = 2\pi i \tilde{C}_1^* \tilde{C}_1.$$

PROOF. (1) and (2) are easy consequences of the fact that

$$XX^\dagger = U_1 U_1^* \text{ and } U_1^* U_1 = I_k.$$

Next, (2) implies that

$$(\lambda I_n - A) U_1 = U_1 (\lambda I_k - \tilde{A}).$$

If  $\lambda \notin \sigma(A)$  then both sides of the preceding equality have rank  $k$ . This serves to justify (3) and leads easily to (4). Assertions (5)-(7) are established in much the same way. Assertion (8) rests on the identity  $X A_1^* = A_2 X$ , which

is obtained from the Riccati equation. (9) is readily seen to be equivalent to (8), and (10) and (11) follow easily from (9).  $\square$

LEMMA 5.2. *If (A1)-(A2) are in force, then*

$$\{u \in \mathbb{C}^n : CA^j u = 0 \text{ for } j = 0, \dots, n-1\} \subset \mathcal{N}_P.$$

PROOF. Let  $\mathcal{Q}$  denote the subspace under consideration. Then, since  $\mathcal{Q}$  is invariant under  $A$ , it admits a basis of Jordan chains of  $A$ . But if, say,

$$Av_j = \alpha v_j + v_{j-1} \text{ for } j = i, \dots, \ell,$$

where  $v_0 = 0$  and  $v_j \in \mathcal{Q}$ , then

$$\begin{aligned} (\bar{\alpha} - \alpha)v_1^* Pv_1 &= v_1^*(A^* P - PA)v_1 \\ &= 2\pi i v_1^*(C^* j_{pq} C)v_1 = 0. \end{aligned}$$

Consequently, as  $\alpha \notin \mathbb{R}$  by assumption (A1) and  $P \geq 0$ , it follows that  $v_1 \in \mathcal{N}_P$ . The next step is to observe that

$$\begin{aligned} (\bar{\alpha} - \alpha)v_2^* Pv_2 &= (\alpha v_2 + v_1)^* Pv_2 - v_2^* P(\alpha v_2 + v_1) \\ &= v_2^*(A^* P - PA)v_2 = 0. \end{aligned}$$

Therefore,  $v_2 \in \mathcal{N}_P$ . A simple inductive argument in this spirit serves to prove that all the vectors  $v_1, \dots, v_\ell$  in this chain belong to  $\mathcal{N}_P$ . The same argument holds for all such Jordan chains.  $\square$

LEMMA 5.3. *If (A1)-(A3) are in force and  $y \in \mathbb{C}^n$ , then*

$$(5.12) \quad CA^j Xy = 0 \text{ for } j = 0, \dots, n-1 \iff Xy = 0$$

PROOF. One direction is selfevident. The other direction follows easily from Lemmas 5.2 and 4.2.  $\square$

LEMMA 5.4. *If (A1)-(A3) are in force, then the pairs  $(\tilde{C}, \tilde{A})$ ,  $(\tilde{C}, \tilde{A}_1)$ ,  $(\tilde{C}, \tilde{A}_2)$  are observable and the pairs  $(\tilde{A}, D\tilde{C}^*)$ ,  $(\tilde{A}_1, D\tilde{C}^*)$  and  $(\tilde{A}_2, D\tilde{C}^*)$  are all controllable.*

PROOF. Suppose first that  $\tilde{C}\tilde{A}^j u = 0$  for  $j = 0, \dots, k-1$  and some vector  $u \in \mathbb{C}^k$ . Then  $\tilde{C}\tilde{A}^j u = 0$  for all nonnegative integers  $j$  and hence, by (2) of Lemma 5.1,

$$\tilde{C}\tilde{A}^j u = CA^j U_1 u = 0$$

for  $j = 0, \dots, n-1$ . Thus, by Lemma 5.3, we must have  $U_1 u = 0$ , which in turn implies that  $u = 0$  since  $U_1$  is left invertible. Therefore, the pair  $(\tilde{C}, \tilde{A})$  is also observable. Moreover, since

$$\tilde{A}_1 = \tilde{A} + 2\pi i D[\tilde{C}_1^* \ 0]\tilde{C} \quad \text{and} \quad \tilde{A}_2 = \tilde{A} - 2\pi i D[0 \ \tilde{C}_2^*]\tilde{C},$$

it follows easily that the pairs  $(\tilde{C}, \tilde{A}_1)$  and  $(\tilde{C}, \tilde{A}_2)$  are also observable.

Next, observability of the pair  $(\tilde{C}, \tilde{A}_2)$  implies the observability of the pair  $(\tilde{C}D, \tilde{A}_1^*)$ , since  $D^{-1}\tilde{A}_2 D = \tilde{A}_1^*$ . Therefore, the pairs  $(\tilde{C}D, \tilde{A}^*)$  and  $(\tilde{C}D, \tilde{A}_2^*)$  are also observable and the pairs  $(\tilde{A}, D\tilde{C}^*)$ ,  $(\tilde{A}_1, D\tilde{C}^*)$  and  $(\tilde{A}_2, D\tilde{C}^*)$  are all controllable.  $\square$

**THEOREM 5.1.** *The realizations*

$$(5.13) \quad \Theta_X(\lambda) = I_m - 2\pi i \tilde{C}(\lambda I_k - \tilde{A})^{-1} D \tilde{C}^* J$$

and

$$(5.14) \quad \Sigma_X(\lambda) = I_m - 2\pi i J \tilde{C}(\lambda I_k - \tilde{A}_2)^{-1} D \tilde{C}^* J$$

are both minimal and

$$\sigma(\widetilde{A}_2) \subset \mathbb{C}_- .$$

**PROOF.** The stated formulas for  $\Theta_X(\lambda)$  and  $\Sigma_X(\lambda)$  follow easily from formulas (5.4) and (5.5) and Lemma 5.1. The asserted minimality of these realizations is a consequence of Lemma 5.4. The last assertion then follows from the fact that  $\Sigma_X \in \mathcal{S}^{m \times m}$  and the exhibited realization is minimal.  $\square$

**5.5. Reproducing Kernel Hilbert Spaces.** A Hilbert space  $\mathcal{H}$  of  $m \times 1$  vector valued functions that are defined on a subset  $\Omega$  of the complex plane  $\mathbb{C}$  is a reproducing kernel Hilbert space if there exists an  $m \times m$  matrix valued function  $K_\omega(\lambda)$  on  $\Omega \times \Omega$  such that for every choice of  $\omega \in \Omega, v \in \mathbb{C}^m$  and  $f \in \mathcal{H}$  the following two conditions prevail:

- (1)  $K_\omega v \in \mathcal{H}$ .
- (2)  $\langle f, K_\omega v \rangle_{\mathcal{H}} = v^* f(\omega)$ .

A matrix valued function  $K_\omega(\lambda)$  that meets these two conditions is termed a reproducing kernel. It is readily checked that a reproducing kernel Hilbert space has exactly one reproducing kernel.

## 6. Description of the solutions to the first subproblem

The objective of this section is to establish the following theorem:

**THEOREM 6.1.** *If assumptions (A1)-(A3) are in force, then*

$$(6.1) \quad \widehat{\mathcal{S}}_X(C, A, P) = \{T_{\Theta_X}[\varepsilon] : \varepsilon \in \mathcal{S}^{p \times q}\} .$$

The proof of this theorem is somewhat lengthy, and is presented in two subsections. In the first (which is technically more challenging) we shall show that the left hand side of formula (6.1) is included in the right hand side; the second subsection is devoted to the opposite inclusion.

**6.1. From left to right.** In this subsection we shall show that if assumptions (A1)-(A3) are in force and  $S \in \widehat{\mathcal{S}}_X(A, A, P)$ , then

$$S = T_{\Theta_X}[\varepsilon]$$

for some  $\varepsilon \in \mathcal{S}^{p \times q}$ . In contrast to the rest of this paper, which requires very little in the way of preparation, here we shall invoke some results from the theory of vector valued Hardy spaces  $H_2^k$  with respect to  $\mathbb{C}_+$  and the theory of reproducing kernel Hilbert spaces of vector valued functions.

To begin with, under assumptions (A1)-(A3), the space

$$(6.2) \quad \mathcal{M}_X = \{F(\lambda)Xu : u \in \mathbb{C}^n\}$$

endowed with the inner product

$$(6.3) \quad \langle FXu, FXv \rangle_{\mathcal{M}_X} = v^* Xu$$

is a finite dimensional reproducing kernel Hilbert space with reproducing kernel

$$(6.4) \quad K_\omega(\lambda) = F(\lambda)XF(\omega)^*.$$

The fact that  $\mathcal{M}_X$  is a Hilbert space follows from Lemma 5.3. Moreover, since  $X$  is a solution of the Riccati equation (4.1), it is readily checked by direct calculation that if  $\Theta_X(\lambda)$  is specified by formula (5.4) and

$$\rho_\omega(\lambda) = -2\pi i(\lambda - \omega^*),$$

then

$$(6.5) \quad \frac{J - \Theta_X(\lambda)J\Theta_X(\omega)^*}{\rho_\omega(\lambda)} = F(\lambda)XF(\omega)^*.$$

The fact that the reproducing kernel of the space  $\mathcal{M}_X$  can also be expressed in the special form

$$(6.6) \quad K_\omega(\lambda) = \frac{J - \Theta_X(\lambda)J\Theta_X(\omega)^*}{\rho_\omega(\lambda)},$$

is not an accident. It is a consequence of a characterization of reproducing kernel Hilbert spaces with reproducing kernels of the form (6.6) (in terms of an invariance condition and a structural identity) that is due to L. de Branges [Br]. For more information on de Branges's theorem in a number of different settings, including extensions to Krein spaces, see [AD] and the references cited therein. For applications of this characterization to the case at hand, see [D5], [D6] and, for an expository introduction to these spaces and additional references, Sections 3-7 of [D7].

Next, the assumption that  $\sigma(A) \cap \mathbb{R} = \emptyset$  insures that if  $S \in \mathcal{S}^{p \times q}$ , and  $u \in \mathbb{C}^n$ , then:

$[I_p - S]Fu$  is holomorphic in  $\mathbb{C}_+$  if and only if it is in the Hardy space  $H_2^p$ , whereas

$[-S^\# I_q]Fu$  is holomorphic in  $\mathbb{C}_-$  if and only if it is orthogonal to the Hardy space  $H_2^q$  with respect to the standard inner product:

$$\langle [-S^\# I_q]Fu, \varphi \rangle = 0 \quad \text{for every } \varphi \in H_2^q.$$

For every  $S \in \mathcal{S}^{p \times q}$ , there is an associated reproducing kernel Hilbert space with reproducing kernel

$$(6.7) \quad k_\omega^S(\lambda) = \frac{I_p - S(\lambda)S(\omega)^*}{\rho_\omega(\lambda)} \quad \lambda, \omega \in \mathbb{C}_+.$$

We shall designate this space by  $\mathcal{H}(S)$  and use the following beautiful characterization of this space that is due to de Branges and Rovnyak [BrR] (written here for  $\mathbb{C}_+$ ).

THEOREM 6.2. Let  $S \in \mathcal{S}^{p \times q}$  and for  $f \in H_2^p$  let

$$\alpha_f = \sup\{\|f + Sg\|^2 - \|g\|^2 : g \in H_2^q\}.$$

Then the RKHS

$$\mathcal{H}(S) = \{f \in H_2^p : \alpha_f < \infty\}$$

endowed with the norm

$$\|f\|_{\mathcal{H}(S)}^2 = \alpha_f.$$

LEMMA 6.1. If assumptions (A1)-(A3) are met and  $S \in \widehat{\mathcal{S}}_X(C, A, P)$ , then

$$[I_p - S]FXy \in \mathcal{H}(S)$$

for every  $y \in \mathbb{C}^n$  and

$$\|[I_p - S]FXy\|_{\mathcal{H}(S)} \leq \|FXy\|_{\mathcal{H}(\Theta_X)}.$$

PROOF. Fix a vector  $y \in \mathbb{C}^n$  and let

$$F(\lambda)Xy = f(\lambda) = \begin{bmatrix} g(\lambda) \\ h(\lambda) \end{bmatrix},$$

where  $g(\lambda) \in \mathbb{C}^p$  and  $h(\lambda) \in \mathbb{C}^q$  for every point  $\lambda \in \mathbb{C} \setminus \sigma(A)$ . In the given setting,

$$g - Sh \in H_2^p \text{ and } -S^\# g + h \text{ is orthogonal to } H_2^q.$$

Therefore, for every  $\varphi \in H_2^q$ ,

$$\|g - Sh + S\varphi\|^2 = \|g - Sh\|^2 + 2\Re\langle g - Sh, S\varphi \rangle + \|S\varphi\|^2.$$

But, now as

$$\begin{aligned} \|g - Sh\|^2 &= \langle g - Sh, g \rangle + \langle g - Sh, -Sh \rangle \\ &= \langle g - Sh, g \rangle + \langle -S^*g + h, h \rangle - \langle (I_q - S^*S)h, h \rangle \\ &= \langle \Delta_S f, f \rangle - \langle (I_q - S^*S)h, h \rangle \end{aligned}$$

and

$$\begin{aligned} \langle g - Sh, S\varphi \rangle &= \langle S^*g - h, \varphi \rangle + \langle (I_q - S^*S)h, \varphi \rangle \\ &= \langle (I_q - S^*S)h, \varphi \rangle, \end{aligned}$$

it is readily seen that

$$\|g - Sh + S\varphi\|^2 - \|\varphi\|^2 = \langle \Delta_S f, f \rangle - \langle (I_q - S^*S)(h - \varphi), (h - \varphi) \rangle$$

for every  $\varphi \in H_2^q$ . Therefore, by Theorem 6.2,  $g - Sh \in \mathcal{H}(S)$  and

$$\begin{aligned} \|g - Sh\|_{\mathcal{H}(S)}^2 &\leq \langle \Delta_S f, f \rangle \\ &= y^* X P_S X y \\ &\leq y^* X P X y \\ &= y^* X y \\ &= \|FXy\|_{\mathcal{H}(\Theta_X)}. \end{aligned}$$

□

LEMMA 6.2. If assumptions (A1)-(A3) are met and  $S \in \widehat{\mathcal{S}}_X(C, A, P)$ , then

$$[I_p - S(\omega)]\Theta_X(\omega)J\Theta_X(\omega)^* \begin{bmatrix} I_p \\ -S(\omega)^* \end{bmatrix} \geq 0$$

for every point  $\omega \in \mathfrak{H}_\Theta \cap \mathbb{C}_+$ .

PROOF. Let  $Z(\lambda) = [I_p - S(\lambda)]$ . Then, in view of Lemma 6.1,

$$\begin{aligned} |v^*Z(\omega)F(\omega)Xy| &= |\langle ZFXy, k_\omega^S v \rangle_{\mathcal{H}(S)}| \\ &\leq \|ZFXy\|_{\mathcal{H}(S)} \|k_\omega^S v\|_{\mathcal{H}(S)} \\ &\leq \|FXy\|_{\mathcal{H}(\Theta_X)} \{v^*k_\omega^S(\omega)v\}^{-1/2} \\ &= \{y^*Xy\}^{1/2} \{v^*k_\omega^S(\omega)v\}^{1/2} \end{aligned}$$

for every point  $\omega \in \mathbb{C}_+ \setminus \sigma(A)$  and every pair of vectors  $y \in \mathbb{C}^n$  and  $v \in \mathbb{C}^p$ . Thus, upon setting

$$y = F(\omega)^*Z(\omega)^*v ,$$

we see that

$$y^*Xy \leq (y^*Xy)^{1/2} \{v^*k_\omega^S(\omega)v\}^{1/2} .$$

But this in turn implies that

$$y^*Xy \leq v^*k_\omega^S(\omega)v ,$$

and hence that

$$v^*Z(\omega)F(\omega)XF(\omega)^*Z(\omega)^*v \leq v^*k_\omega^S(\omega)v .$$

Consequently, upon invoking formulas (6.5) and (6.7), this reduces to

$$v^*Z(\omega)\{J - \Theta_X(\omega)J(\Theta)_X(\omega)^*\}Z(\omega)^*v \leq v^*Z(\omega)JZ(\omega)^*v ,$$

which leads easily to the stated conclusion of the lemma.  $\square$

LEMMA 6.3. If assumptions (A1)-(A3) are in force, then

$$(6.8) \quad \widehat{\mathcal{S}}_X(C, A, P) \subseteq \{T_{\Theta_X}[\varepsilon] : \varepsilon \in \mathcal{S}^{p \times q}\} .$$

PROOF. Let  $S \in \widehat{\mathcal{S}}_X(C, A, P)$  and let

$$[I_p - S(\omega)]\Theta_X(\omega) = [M(\omega) \ N(\omega)]$$

for every point  $\omega \in \mathfrak{H}_{\Theta_X} \cap \mathbb{C}_+$ , where  $M(\omega)$  and  $N(\omega)$  are mvf's of sizes  $p \times p$  and  $p \times q$ , respectively. Then, since

$$\begin{aligned} \det\{\Theta_X(\omega)\} &= \det\{I_m - 2\pi i C(\omega I_n - A)^{-1} X C^* J\} \\ &= \det\{I_n - (\omega I_n - A)^{-1} 2\pi i X C^* J C\} \\ &= \det\{(\omega I_n - A)^{-1}\} \det\{\omega I_n - A - 2\pi i X C^* J C\} , \end{aligned}$$

we see that  $\Theta_X(\omega)$  is an invertible  $m \times m$  matrix except for at most a finite number of points. Thus, the  $p \times m$  mvf  $[M(\omega) \ N(\omega)]$  has rank  $p$  in  $\mathbb{C}_+$  except for at most a finite number of points. Moreover, since the previous lemma implies that

$$M(\omega)M(\omega)^* - N(\omega)N(\omega)^* \geq 0$$

in  $\mathfrak{H}_{\Theta_X} \cap \mathbb{C}_+$ , it follows readily that  $M(\omega)$  is invertible in  $\mathbb{C}_+$  except for at most a finite number of points and that  $M^{-1}N \in \mathcal{S}^{p \times q}$ . Thus, upon setting  $M^{-1}N = -\varepsilon$  and writing this equality out in terms of the blocks  $\Theta_{ij}$  of  $\Theta_X$ , we get

$$\Theta_{12} - S\Theta_{22} = (\Theta_{11} - S\Theta_{21})(-\varepsilon)$$

and hence that

$$S = T_{\Theta_X}[\varepsilon] ,$$

as needed.  $\square$

**6.2. From right to left.** In this subsection we establish the opposite inclusion.

LEMMA 6.4. *If assumptions (A1)-(A3) are in force and  $\Sigma(\lambda) = \Sigma_X(\lambda)$ , then the following identities hold:*

1.  $[I_q - \Sigma_{12}(\lambda)]F(\lambda)U_1 = \widetilde{C}_1(\lambda I_k - \widetilde{A}_2)^{-1}$ .
2.  $\Sigma_{22}(\lambda)C_2(\lambda I_n - A)^{-1}U_1 = \widetilde{C}_2(\lambda I_k - \widetilde{A}_2)^{-1}$ .
3.  $[-(\Sigma_{12})^\#(\lambda) - I_q]F(\lambda)U_1 = \widetilde{C}_2(\lambda I_k - \widetilde{A}_1)^{-1}$ .
4.  $\Sigma_{11}^\#(\lambda)C_1(\lambda I_n - A)^{-1}U_1 = \widetilde{C}_1(\lambda I_k - \widetilde{A}_1)^{-1}$ .

PROOF. By formula (5.14),

$$\Sigma_{12}(\lambda) = 2\pi i \widetilde{C}_1(\lambda I_k - \widetilde{A}_2)^{-1} D \widetilde{C}_2^*,$$

whereas, by (4) of Lemma 5.1,

$$(6.9) \quad F(\lambda)U_1 = \widetilde{C}(\lambda I_k - \widetilde{A})^{-1} .$$

Thus,

$$[I_p - \Sigma_{12}(\lambda)]F(\lambda)U_1 = \widetilde{C}_1(\lambda I_k - \widetilde{A}_2)^{-1}(\lambda I_k - \widetilde{A}_2 - 2\pi i D \widetilde{C}_1^* \widetilde{C}_1)(\lambda I_k - \widetilde{A})^{-1} ,$$

which is easily seen to yield (1). Next, since

$$\Sigma_{22}(\lambda) = I_q - 2\pi i \widetilde{C}_2(\lambda I_k - \widetilde{A}_2)^{-1} D \widetilde{C}_2^*$$

and

$$C_2(\lambda I_n - A)^{-1}U_1 = \widetilde{C}_2(\lambda I_k - \widetilde{A})^{-1} ,$$

it is readily seen that

$$\Sigma_{22}(\lambda)C_2(\lambda I_n - A)^{-1}U_1 = \widetilde{C}_2(\lambda I_k - \widetilde{A}_2)^{-1}(\lambda I_k - \widetilde{A}_2 - 2\pi i D \widetilde{C}_2^* \widetilde{C}_2)(\lambda I_k - \widetilde{A})^{-1} ,$$

which reduces to (2).

The proof of (3) is much the same as (1) once you observe that

$$(\Sigma_{12})^\#(\lambda) = -2\pi i \widetilde{C}_2 D(\lambda I_k - \widetilde{A}_2^*)^{-1} \widetilde{C}_1^* = -2\pi i \widetilde{C}_2(\lambda I_k - \widetilde{A}_1)^{-1} D \widetilde{C}_1^* .$$

Similarly, the proof of (4) is much the same as the proof of (2) once you observe that

$$\Sigma_{11}^\#(\lambda) = I_p + 2\pi i \widetilde{C}_1 D(\lambda I_k - \widetilde{A}_2^*)^{-1} \widetilde{C}_1^* = I_p + 2\pi i \widetilde{C}_1(\lambda I_k - \widetilde{A}_1)^{-1} D \widetilde{C}_1^* .$$

$\square$

LEMMA 6.5. *If assumptions (A1)-(A3) are in force, then*

$$\|\Sigma_{21}(\mu)\|^2 \leq 1 - |\det \Sigma_{22}(\mu)|^2$$

for every point  $\mu \in \mathbb{R}$ .

PROOF. The identity  $\Sigma(\mu)\Sigma(\mu)^* = I_m$  for  $\mu \in \mathbb{R}$  implies in particular that

$$\Sigma_{21}(\mu)\Sigma_{21}(\mu)^* + \Sigma_{22}(\mu)\Sigma_{22}^*(\mu) = I_q$$

for every point  $\mu \in \mathbb{R}$ . Let  $s_1(\mu) \geq \dots \geq s_q(\mu)$  denote the singular values of  $\Sigma_{22}(\mu)^*$ . Then, for every vector  $x \in \mathbb{C}^q$ ,

$$\|\Sigma_{22}(\mu)^*x\|^2 \geq s_q(\mu)^2\|x\|^2 \geq |\det \Sigma_{22}(\mu)|^2\|x\|^2 ,$$

since  $s_j(\mu) \leq 1$  for  $j = 1, \dots, q$ . Thus,

$$\|\Sigma_{21}(\mu)^*x\|^2 = \|x\|^2 - \|\Sigma_{22}(\mu)^*x\|^2 \leq \{1 - |\det \Sigma_{22}(\mu)|^2\}\|x\|^2 ,$$

which clearly exhibits the validity of the asserted inequality.  $\square$

LEMMA 6.6. *If assumptions (A1)-(A3) are in force, then there exists a  $\delta > 0$  such that*

$$|\det\{\Sigma_{22}(\mu)\}| \geq \delta$$

for every point  $\mu \in \mathbb{R}$ .

PROOF. By formula (5.14),

$$\Sigma_{22}(\lambda) = I_q - 2\pi i \widetilde{C}_2(\lambda I_k - \widetilde{A}_2)^{-1} D \widetilde{C}_2^*$$

for every point  $\lambda \in \mathbb{C}_+$ . Therefore,

$$\begin{aligned} \det\{\Sigma_{22}(\lambda)\} &= \det\{I_p - 2\pi i \widetilde{C}_2(\lambda I_k - \widetilde{A}_2)^{-1} D \widetilde{C}_2^*\} \\ &= \det\{I_p - 2\pi i(\lambda I_k - \widetilde{A}_2)^{-1} D \widetilde{C}_2^* \widetilde{C}_2\} \\ &= \det(\lambda I_k - \widetilde{A}_2)^{-1} \det(\lambda I_k - \widetilde{A}) . \end{aligned}$$

This exhibits  $\det\{\Sigma_{22}(\lambda)\}$  as the ratio of two monic polynomials of degree  $k$ . Moreover, since  $\sigma(\widetilde{A}_2) \subset \mathbb{C}_-$  and  $\sigma(\widetilde{A}) \cap \mathbb{R} = \emptyset$ , it follows that

$$|\det\{\Sigma_{22}(\mu)\}| > 0 \text{ for } \mu \in \mathbb{R}$$

and

$$|\det \Sigma_{22}(\mu)| \rightarrow 1 \text{ as } \mu \rightarrow \pm\infty .$$

Therefore, since  $\det\{\Sigma_{22}(\mu)\}$  is clearly a continuous function of  $\mu \in \mathbb{R}$ , there must exist a  $\delta > 0$  which meets the asserted bound.  $\square$

THEOREM 6.3. *If assumptions (A1)-(A3) are in force, then*

$$\|\Sigma_{21}(\lambda)\|^2 \leq 1 - \delta^2$$

for every point  $\lambda \in \overline{\mathbb{C}_+}$ .

PROOF. The preceding two lemmas guarantee the existence of a  $\delta > 0$  such that

$$\|\Sigma_{21}(\mu)\|^2 \leq 1 - \delta^2$$

for every point  $\mu \in \mathbb{R}$ . Therefore,

$$|\xi^* \Sigma_{21}(\lambda) \eta| \leq \|\Sigma_{21}(\lambda)\| |\xi| \|\eta\| \leq (1 - \delta^2)^{1/2} \|\xi\| \|\eta\|$$

for  $\lambda \in \mathbb{R}$ . Moreover, since  $\xi^* \Sigma_{21}(\lambda) \eta$  is holomorphic in  $\overline{\mathbb{C}_+}$  and the same bound is clearly in effect for all points  $\lambda \in \overline{\mathbb{C}_+}$  with  $|\lambda| \geq R$  for large enough  $R$ , it must hold for all points  $\overline{\mathbb{C}_+}$  by the maximum principle. Now choose  $\xi = \Sigma_{21}(\lambda) \eta$  to obtain

$$\|\Sigma_{21}(\lambda) \eta\|^2 \leq (1 - \delta^2)^{1/2} \|\Sigma_{21}(\lambda) \eta\| \|\eta\|.$$

But this implies that

$$\|\Sigma_{21}(\lambda) \eta\| \leq (1 - \delta^2)^{1/2} \|\eta\|,$$

which is clearly equivalent to the stated bound.  $\square$

LEMMA 6.7. *If (A1)-(A3) are in force and  $\lambda \notin \sigma(\tilde{A}^*) \cup \sigma(\widetilde{A_1}) \cup \sigma(\widetilde{A_2})$ , then*

$$\begin{aligned} & (\lambda I_k - \tilde{A}^*)^{-1} \{ \widetilde{C_1}^* \widetilde{C_1} (\lambda I_k - \widetilde{A_2})^{-1} + \widetilde{C_2}^* \widetilde{C_2} (\lambda I_k - \widetilde{A_1})^{-1} \} \\ &= \frac{D^{-1}}{2\pi i} \{ (\lambda I_k - \widetilde{A_1})^{-1} - (\lambda I_k - \widetilde{A_2})^{-1} \}. \end{aligned}$$

PROOF. This is a straightforward calculation based on Items 10 and 11 in Lemma 5.1.  $\square$

LEMMA 6.8. *If assumptions (A1)-(A3) are in force, then*

$$(6.10) \quad \int_{-\infty}^{\infty} \{ U_1^* F(\mu)^* \Delta_{\Sigma_{12}}(\mu) F(\mu) U_1 \} d\mu = D^{-1}$$

PROOF. By Lemma 6.4,

$$\Delta_{\Sigma_{12}}(\mu) F(\mu) U_1 = \begin{bmatrix} \widetilde{C_1}(\mu I_k - \widetilde{A_2})^{-1} \\ \widetilde{C_2}(\mu I_k - \widetilde{A_1})^{-1} \end{bmatrix}.$$

Therefore, the integrand in formula (6.10) can be written as

$$\{\dots\} = (\mu I_k - \tilde{A}^*)^{-1} \{ \widetilde{C_1}^* \widetilde{C_1} (\mu I_k - \widetilde{A_2})^{-1} + \widetilde{C_2}^* \widetilde{C_2} (\mu I_k - \widetilde{A_1})^{-1} \}.$$

Thus, in view of Lemma 6.7, we can write the integral of interest as

$$\lim_{\varepsilon \downarrow 0} \frac{1}{2\pi i} \int_{-\infty}^{\infty} D^{-1} \{ (\mu I_k - \widetilde{A_1})^{-1} - (\mu I_k - \widetilde{A_2})^{-1} \} \frac{d\mu}{1 + i\varepsilon\mu}.$$

Next, since  $\sigma(\widetilde{A_1}) \subset \mathbb{C}_+$ ,  $\sigma(\widetilde{A_2}) \subset \mathbb{C}_-$  and we have chosen  $\varepsilon > 0$ , we can invoke Cauchy's theorem to obtain the evaluations:

$$\begin{aligned} \frac{1}{2\pi i} \int_{-\infty}^{\infty} (\mu I_k - \widetilde{A_1})^{-1} \frac{d\mu}{1 + i\varepsilon\mu} &= \lim_{R \uparrow \infty} \frac{1}{2\pi i} \int_{-R}^R (\mu I_k - \widetilde{A_1})^{-1} \frac{d\mu}{i\varepsilon(\mu - i/\varepsilon)} \\ &= 0 \end{aligned}$$

and

$$\begin{aligned} \frac{1}{2\pi i} \int_{-\infty}^{\infty} (\mu I_k - \widetilde{A}_2)^{-1} \frac{d\mu}{1 + i\varepsilon\mu} &= \lim_{R \uparrow \infty} \frac{1}{2\pi i} \int_{-R}^R (\mu I_k - \widetilde{A}_2)^{-1} \frac{d\mu}{i\varepsilon(\mu - i/\varepsilon)} \\ &= \frac{1}{i\varepsilon} ((i/\varepsilon)I_k - \widetilde{A}_2)^{-1} \\ &= -(I_k + i\varepsilon\widetilde{A}_2)^{-1}. \end{aligned}$$

Formula (6.10) now follows easily upon combining all the calculations, since

$$\lim_{\varepsilon \downarrow 0} (I_k + i\varepsilon\widetilde{A}_2)^{-1} = I_k.$$

□

**LEMMA 6.9.** *If (A1)-(A3) are in force and  $S = T_{\Theta_X}[\varepsilon]$  for some mvf  $\varepsilon \in \mathcal{S}^{p \times q}$ , then*

$$(6.11) \quad \int_{-\infty}^{\infty} U_1^* F(\mu)^* \Delta_S(\mu) F(\mu) U_1 d\mu = D^{-1}.$$

PROOF. Let

$$T(\lambda) = S(\lambda) - \Sigma_{12}(\lambda).$$

Then, by formulas (6.9) and (5.3),

$$\begin{aligned} &U_1^* F(\mu)^* \{ \Delta_S(\mu) - \Delta_{\Sigma_{12}}(\mu) \} F(\mu) U_1 \\ &= (\mu I_k - \widetilde{A}^*)^{-1} [\widetilde{C}_1^* \quad \widetilde{C}_2^*] \begin{bmatrix} 0 & -T(\mu) \\ -T^*(\mu) & 0 \end{bmatrix} \begin{bmatrix} \widetilde{C}_1 \\ \widetilde{C}_2 \end{bmatrix} (\mu I_k - \widetilde{A})^{-1} \\ &= \alpha(\mu) + \beta(\mu) \end{aligned}$$

for  $\mu \in \mathbb{R}$ , where

$$\alpha(\mu) = -(\mu I_k - \widetilde{A}^*)^{-1} \widetilde{C}_1^* T(\mu) \widetilde{C}_2 (\mu I_k - \widetilde{A})^{-1}$$

and

$$\beta(\mu) = -(\mu I_k - \widetilde{A}^*)^{-1} \widetilde{C}_2^* T^*(\mu) \widetilde{C}_1 (\mu I_k - \widetilde{A})^{-1}.$$

Therefore, since

$$T(\mu) = \Sigma_{11}(\mu) W_{\varepsilon}(\mu) \Sigma_{22}(\mu),$$

where

$$W_{\varepsilon}(\lambda) = \varepsilon(\lambda) \{ I_q - \Sigma_{21}(\lambda) \varepsilon(\lambda) \}^{-1},$$

is a bounded holomorphic mvf in  $\mathbb{C}_+$ , we can reexpress  $\alpha(\mu)$  and  $\beta(\mu)$  as

$$\alpha(\mu) = -(\mu I_k - \widetilde{A}_1^*)^{-1} \widetilde{C}_1^* W_{\varepsilon}(\mu) \widetilde{C}_2 (\mu I_k - \widetilde{A}_2)^{-1}$$

and

$$\beta(\mu) = -(\mu I_k - \widetilde{A}_2^*)^{-1} \widetilde{C}_2^* W_{\varepsilon}^*(\mu) \widetilde{C}_1 (\mu I_k - \widetilde{A}_1)^{-1} = \alpha(\mu)^*,$$

respectively. Thus, in order to complete the proof, it suffices to show

$$(6.12) \quad \int_{-\infty}^{\infty} \alpha(\mu) d\mu = 0.$$

If  $\varepsilon(\lambda)$  is analytic in  $\overline{\mathbb{C}_+}$ , then this follows easily from the classical Cauchy theorem since  $\sigma(\widetilde{A}_2) \subset \mathbb{C}_-$  and  $\sigma(\widetilde{A}_1^*) \subset \mathbb{C}_-$ . If not, i.e., if we only know that  $\varepsilon \in \mathcal{S}^{p \times q}$ , then formula (6.12) follows from a stronger form of Cauchy's theorem that is applicable to functions that belong to the Hardy space  $H_2$ ; see e.g., Theorem 5.19 in [RR].  $\square$

**THEOREM 6.4.** *If assumptions (A1)-(A3) are in force and  $S = T_{\Theta_X}[\varepsilon]$  for some  $\varepsilon \in \mathcal{S}^{p \times q}$ , then  $S \in \widehat{\mathcal{S}}_X(C, A, P)$ .*

**PROOF.** It is convenient to use the Redheffer form (5.3) and to keep in mind that  $\mathcal{R}_X = \mathcal{R}_{U_1}$ . We proceed in steps.

1.  $[I_p - \Sigma_{12}(\lambda)]F(\lambda)X$  is holomorphic in  $\mathbb{C}_+$ .

This is clear from (1) of Lemma 6.4, since  $\sigma(\widetilde{A}_2) \subset \mathbb{C}_-$ .

2.  $[0 - \Sigma_{11}(\lambda)\varepsilon(\lambda)(I_q - \Sigma_{21}(\lambda)\varepsilon(\lambda))^{-1}\Sigma_{22}(\lambda)]F(\lambda)X$  is holomorphic in  $\mathbb{C}_+$ .

This is clear from (2) of Lemma 6.4 and Theorem 6.3. The latter guarantees that  $\{I_q - \Sigma_{21}(\lambda)\varepsilon(\lambda)\}^{-1}$  is holomorphic in  $\mathbb{C}_+$ .

The verification that  $[-S^\# \quad I_q]F(\lambda)X$  is holomorphic in  $\mathbb{C}_-$  follows in much the same way from (3) and (4) of Lemma 6.4 and another application of Theorem 6.3. (It is more convenient to prove that  $U_1^*F^\#(\lambda) \begin{bmatrix} -S(\lambda) \\ I_q \end{bmatrix}$  is holomorphic in  $\mathbb{C}_+$ .)

Finally, by formulas (6.11) and (5.6),

$$\int_{-\infty}^{\infty} \{XF(\mu)^*\Delta_S(\mu)F(\mu)X\}d\mu = U_1 D D^{-1} D U_1^* = X .$$

$\square$

## 7. The second problem

The next order of business is to identify the set

$$\widehat{\mathcal{S}}_X(C, A, P) \cap \widehat{\mathcal{S}}_N(C, A, P) .$$

**LEMMA 7.1.** *If (A1)-(A3) are in force and  $S = T_{\Theta}[\varepsilon]$  for some  $\varepsilon \in \mathcal{S}^{p \times q}$ , then*

$$(7.1) \quad [I_p - S(\lambda)] = \Sigma_{11}(\lambda)\{I_p - \varepsilon\Sigma_{21}(\lambda)\}^{-1}[I_p - \varepsilon(\lambda)]\Theta^\#(\lambda)J$$

for  $\lambda \in \mathfrak{H}_{\theta^\#} \cap \mathbb{C}_+$  and

$$(7.2) \quad \begin{bmatrix} -S(\lambda) \\ I_q \end{bmatrix} = -J\Theta(\lambda) \begin{bmatrix} \varepsilon(\lambda) \\ I_q \end{bmatrix} \{I_q - \Sigma_{21}(\lambda)\varepsilon(\lambda)\}^{-1}\Sigma_{22}(\lambda)$$

for  $\lambda \in \mathfrak{H}_\Theta \cap \mathbb{C}_+$ .

**PROOF.** The formulas

$$\Theta^\#(\lambda)J(\Theta)(\lambda) = J ,$$

$$\Theta(\lambda) = \begin{bmatrix} I_p & \Sigma_{12}(\lambda) \\ 0 & I_q \end{bmatrix} \begin{bmatrix} \Sigma_{11}(\lambda) & 0 \\ 0 & \Theta_{22}(\lambda) \end{bmatrix} \begin{bmatrix} I_p & 0 \\ -\Sigma_{21}(\lambda) & I_q \end{bmatrix}$$

and

$$\Theta(\lambda)^{-1} = \begin{bmatrix} I_p & 0 \\ \Sigma_{21}(\lambda) & I_q \end{bmatrix} \begin{bmatrix} \Sigma_{11}(\lambda)^{-1} & 0 \\ 0 & \Sigma_{22}(\lambda) \end{bmatrix} \begin{bmatrix} I_p & -\Sigma_{12}(\lambda) \\ 0 & I_q \end{bmatrix}$$

lead readily to the conclusion that

$$\Theta_{11}^{\#}(\lambda)^{-1} = \Sigma_{11}(\lambda) \text{ and } -(\Theta_{12})^{\#}(\lambda)\Theta_{11}^{\#}(\lambda)^{-1} = \Sigma_{21}(\lambda).$$

Therefore,

$$\{\Theta_{11}^{\#}(\lambda) + \varepsilon(\lambda)(\Theta_{12})^{\#}(\lambda)\}^{-1} = \Sigma_{11}(\lambda)\{I_p - \varepsilon(\lambda)\Sigma_{21}(\lambda)\}^{-1}$$

and hence, by formula (5.2),

$$\begin{aligned} [I_p &- S(\lambda)] &= \Sigma_{11}(\lambda)\{I_p - \varepsilon(\lambda)\Sigma_{21}(\lambda)\}^{-1} \\ &\times [\Theta_{11}^{\#}(\lambda) + \varepsilon(\lambda)(\Theta_{12})^{\#}(\lambda) &- (\Theta_{21})^{\#}(\lambda) - \varepsilon(\lambda)\Theta_{22}^{\#}(\lambda)], \end{aligned}$$

which is equivalent to the first asserted formula.

Similarly, the second formula follows easily from (5.1):

$$\begin{bmatrix} -S(\lambda) \\ I_q \end{bmatrix} = \begin{bmatrix} -\{\Theta_{11}(\lambda)\varepsilon(\lambda) + \Theta_{12}(\lambda)\} \\ \Theta_{21}(\lambda)\varepsilon(\lambda) + \Theta_{22}(\lambda) \end{bmatrix} \{\Theta_{21}(\lambda)\varepsilon(\lambda) + \Theta_{22}(\lambda)\}^{-1},$$

since

$$\{\Theta_{21}(\lambda)\varepsilon(\lambda) + \Theta_{22}(\lambda)\} = \Theta_{22}(\lambda)\{I_q - \Sigma_{21}(\lambda)\varepsilon(\lambda)\}.$$

□

**LEMMA 7.2.** *If assumptions (A1)-(A3) are in force and  $\Theta(\lambda) = \Theta_X(\lambda)$ , then*

$$(7.3) \quad \Theta^{\#}(\lambda)JF(\lambda) = JC\{(I_n - XP)(\lambda I_n - A)^{-1} + X(\lambda I_n - A^*)^{-1}P\}.$$

**PROOF.** By formulas (5.4) and (1.1),

$$\begin{aligned} \Theta^{\#}(\lambda)JC &= \{I_m + 2\pi i J C X (\lambda I_n - A^*)^{-1} C^*\} J C \\ &= J C \{I_n + 2\pi i X (\lambda I_n - A^*)^{-1} C^* J C\} \\ &= J C \{I_n + X (\lambda I_n - A^*)^{-1} (A^* P - P A)\} \\ &= J C \{I_n + X (\lambda I_n - A^*)^{-1} ((A^* - \lambda I_n) P - P (A - \lambda I_n))\} \\ &= J C \{I_n - X P + X (\lambda I_n - A^*)^{-1} P (\lambda I_n - A)\}. \end{aligned}$$

The asserted formula now follows easily upon multiplying both sides of the last expression by  $(\lambda I_n - A)^{-1}$  on the right. □

**LEMMA 7.3.** *If (A1)-(A3) are in force and  $S = T_{\Theta}[\varepsilon]$  for  $\Theta(\lambda) = \Theta_X(\lambda)$  and some  $\varepsilon \in \mathcal{S}^{p \times q}$ , then*

$$[I_p &- S(\lambda)]F(\lambda)u = 0$$

for every point  $\lambda \in \mathbb{C}_+$  and every vector  $u \in \mathcal{N}_P$  if and only if

$$(7.4) \quad [I_p &\varepsilon(\lambda)]JCu = 0$$

for every point  $\lambda \in \mathbb{C}_+$  and every vector  $u \in \mathcal{N}_P$ .

PROOF. Fix a point  $\lambda \in \mathbb{C}_+ \setminus \sigma(A)$  at which  $\Sigma_{11}(\lambda)$  is invertible. Then, in view of formulas (7.1) and (7.3),

$$[I_p \quad -S(\lambda)]F(\lambda)u = 0$$

for every vector  $u \in \mathcal{N}_P$  if and only if

$$[I_p \quad \varepsilon(\lambda)]JC(I_n - XP)(\lambda I_n - A)^{-1}u = 0.$$

Therefore, since

$$\begin{aligned} (I_n - XP)(\lambda I_n - A)^{-1}U_1 &= (I_n - XP)U_1(\lambda I_n - \tilde{A})^{-1} \\ &= (I_n - XP)XU_1D^{-1}(\lambda I_n - \tilde{A})^{-1} \\ &= 0, \end{aligned}$$

we see that

$$[I_p \quad \varepsilon(\lambda)]JC(I_n - XP)(\lambda I_n - A)^{-1}u = 0$$

for every  $u \in \mathcal{N}_P$  if and only if

$$[I_p \quad \varepsilon(\lambda)]JC(I_n - XP)(\lambda I_n - A)^{-1}(u + Xy) = 0$$

for every  $u \in \mathcal{N}_P$  and every  $y \in \mathbb{C}^n$ . But clearly this last condition holds if and only if

$$[I_p \quad \varepsilon(\lambda)]JC(I_n - XP) = 0,$$

or, equivalently, if and only if

$$[I_p \quad \varepsilon(\lambda)]JCu = 0$$

for every vector  $u \in \mathcal{N}_P$ . This gives the desired conclusion for all points  $\lambda \in \mathbb{C}_+ \setminus \sigma(A)$  at which  $\Sigma_{11}(\lambda)$  is invertible, that is to say for all points  $\lambda \in \mathbb{C}_+$  with the possible exception of at most finitely many points. Therefore, the conclusion must hold for all points  $\lambda \in \mathbb{C}_+$ .  $\square$

LEMMA 7.4. *If (A1)-(A3) are in force and  $S = T_\Theta[\varepsilon]$  for  $\Theta(\lambda) = \Theta_X(\lambda)$  and some mvf  $\varepsilon \in \mathcal{S}^{p \times q}$ , then*

$$[-S^\#(\lambda) \quad I_q]F(\lambda)u = 0$$

for every point  $\lambda \in \mathbb{C}_-$  and every vector  $u \in \mathcal{N}_P$  if and only if

$$(7.5) \quad u^*C^*J \begin{bmatrix} \varepsilon(\lambda) \\ I_q \end{bmatrix} = 0$$

for every point  $\lambda \in \mathbb{C}_+$  and every vector  $u \in \mathcal{N}_P$ .

PROOF. The condition of interest is equivalent to the statement that

$$u^*F^\#(\lambda) \begin{bmatrix} -S(\lambda) \\ I_q \end{bmatrix} = 0$$

for every point  $\lambda \in \mathbb{C}_+$  and every vector  $u \in \mathcal{N}_P$ . Fix a point  $\lambda \in \mathbb{C}_+ \setminus \sigma(A^*)$  at which  $\Sigma_{22}(\lambda)$  is invertible. Then, in view of formula (7.2),

$$u^*F^\#(\lambda) \begin{bmatrix} -S(\lambda) \\ I_q \end{bmatrix} = 0$$

for every  $u \in \mathcal{N}_P$  if and only if

$$u^* F^\#(\lambda) J \Theta(\lambda) \begin{bmatrix} \varepsilon(\lambda) \\ I_q \end{bmatrix} = 0$$

for every  $u \in \mathcal{N}_P$ . By Lemma 7.2,

$$\begin{aligned} u^* F^\#(\lambda) J \Theta(\lambda) &= u^* \{(\lambda I_n - A^*)^{-1}(I_n - PX) + P(\lambda I_n - A)^{-1}X\} C^* J \\ &= u^* (\lambda I_n - A^*)^{-1}(I_n - PX) C^* J, \end{aligned}$$

and the asserted conclusion now follows much as in the proof of Lemma 7.3.  $\square$

**LEMMA 7.5.** *If assumptions (A1)-(A3) are in force, then  $\{Cu : u \in \mathcal{N}_P\}$  is a  $J$ -neutral subspace of  $\mathbb{C}^m$ .*

**PROOF.** It is clear that the indicated set of vectors is a subspace of  $\mathbb{C}^m$ . If  $u, v \in \mathcal{N}_P$ , then it follows readily from the Lyapunov equation (1.1) that

$$\begin{aligned} (Cu)^* J Cv &= u^* C^* J Cv \\ &= (2\pi i)^{-1} u^* (A^* P - PA)v \\ &= 0. \end{aligned}$$

$\square$

Let  $w_1, \dots, w_\nu$  be an orthogonal basis for the  $J$ -neutral subspace  $\{Cu : u \in \mathcal{N}_P\}$  such that

$$\|w_j\|^2 = 2, \text{ for } j = 1, \dots, \nu.$$

Then

$$w_j^* J w_i = 0, \text{ for } i, j = 1, \dots, \nu$$

and

$$w_j^* w_i = 0, \text{ for } i, j = 1, \dots, \nu \text{ and } i \neq j.$$

Hence, upon writing

$$w_j^* = [x_j^* \quad y_j^*]$$

with  $x_j \in \mathbb{C}^p$  and  $y_j \in \mathbb{C}^q$ , we see that

$\{x_j\}, j = 1, \dots, \nu$ , is an orthonormal family in  $\mathbb{C}^p$

and

$\{y_j\}, j = 1, \dots, \nu$ , is an orthonormal family in  $\mathbb{C}^q$ .

Therefore

$$(7.6) \quad \nu \leq \min\{p, q\}.$$

Moreover, the condition (7.4) holds, if and only if

$$(7.7) \quad x_j = \varepsilon(\lambda) y_j, \text{ for } \lambda \in \mathbb{C}_+ \text{ and } j = 1, \dots, \nu,$$

whereas condition (7.5) holds if and only if

$$(7.8) \quad x_j^* \varepsilon(\lambda) = y_j^*, \text{ for } \lambda \in \mathbb{C}_+ \text{ and } j = 1, \dots, \nu.$$

However, since  $\varepsilon \in \mathcal{S}^{p \times q}$ , it follows from the maximum principle that these two conditions are equivalent, see e.g., the discussion on pages 7-9 of [D1] for a more detailed explanation, if need be.

Let  $U = [U_1 \ U_2]$  be a  $p \times p$  unitary matrix with  $U_2 = [x_1 \cdots x_\nu]$  and let  $V = [V_1 \ V_2]$  be a  $q \times q$  unitary matrix with  $V_2 = [y_1 \cdots y_\nu]$ . Then the condition (7.7) holds if and only if

$$U_2 = \varepsilon(\lambda)V_2, \text{ for } \lambda \in \mathbb{C}_+,$$

whereas the equivalent second condition (7.8) holds if and only if

$$U_2^*\varepsilon(\lambda) = V_2^*, \text{ for } \lambda \in \mathbb{C}_+.$$

Thus, under condition (7.7),

$$\begin{aligned} \varepsilon(\lambda) &= UU^*\varepsilon(\lambda)VV^* \\ &= U \begin{bmatrix} U_1^*\varepsilon(\lambda)V_1 & U_1^*\varepsilon(\lambda)V_2 \\ U_2^*\varepsilon(\lambda)V_1 & U_2^*\varepsilon(\lambda)V_2 \end{bmatrix} V^* \\ &= U \begin{bmatrix} U_1^*\varepsilon(\lambda)V_1 & U_1^*U_2 \\ V_2^*V_1 & V_2^*V_2 \end{bmatrix} V^*, \end{aligned}$$

which reduces to

$$(7.9) \quad \varepsilon(\lambda) = U \begin{bmatrix} \tilde{\varepsilon}(\lambda) & 0 \\ 0 & I_\nu \end{bmatrix} V^*,$$

where

$$\tilde{\varepsilon}(\lambda) = U_1^*\varepsilon(\lambda)V_1$$

is an arbitrary element in  $\mathcal{S}^{(p-\nu) \times (q-\nu)}$ .

Conversely, if formula (7.9) is used to define  $\varepsilon(\lambda)$  for any choice of  $\tilde{\varepsilon} \in \mathcal{S}^{(p-\nu) \times (q-\nu)}$ , then  $\varepsilon \in \mathcal{S}^{p \times q}$  and meets the requirement (7.7).

Thus, the analysis of the preceding sections, combined with the calculations in this section, serves to establish most of the following result:

**THEOREM 7.1.** *If assumptions (A1)-(A3) are in force, then there exist a pair of unitary matrices  $U \in \mathbb{C}^{p \times p}$  and  $V \in \mathbb{C}^{q \times q}$  such that*

$$(7.10) \quad \widehat{\mathcal{S}}(C, A, P) = \left\{ T_{\Theta_X} \left[ U \begin{bmatrix} \tilde{\varepsilon} & 0 \\ 0 & I_\nu \end{bmatrix} V^* \right] : \tilde{\varepsilon} \in \mathcal{S}^{(p-\nu) \times (q-\nu)} \right\}.$$

Moreover,

$$(7.11) \quad \nu = \text{rank}\{P + C_1^*C_1\} - \text{rank}P = \text{rank}\{P + C_2^*C_2\} - \text{rank}P,$$

where  $C_1$  and  $C_2$  are the top and bottom block components of  $C$  of sizes  $p \times n$  and  $q \times n$ , respectively (i.e.,  $C^* = [C_1^* \ C_2^*]$ ).

**PROOF.** To obtain formula (7.11), let  $L$  denote the linear operator that maps  $u \in \mathcal{N}_P$  into  $Cu \in \mathbb{C}^m$ . Then the dimension of the nullspace  $\mathcal{N}_L$  of  $L$  is related to  $\nu$  by the standard formula

$$\dim \mathcal{N}_P = \nu + \dim \mathcal{N}_L.$$

Thus, as

$$\mathcal{N}_L = \mathcal{N}_C \cap \mathcal{N}_P = \mathcal{N}_{P+C^*C}$$

and, as follows from the Lyapunov equation (1.1),

$$\mathcal{N}_{P+C^*C} = \mathcal{N}_{P+C_1^*C_1} = \mathcal{N}_{P+C_2^*C_2},$$

it is readily seen that

$$\nu = \dim \mathcal{N}_P - \dim \mathcal{N}_{P+C_1^*C_1} = \dim \mathcal{N}_P - \dim \mathcal{N}_{P+C_2^*C_2}.$$

The last two formulas are equivalent to the formulas in (7.11). Everything else is covered by the preceding discussion.  $\square$

## 8. The final touch

One of the goals of this paper was to obtain a description of the set  $\widehat{\mathcal{S}}(C, A, P)$  under assumptions (A1)-(A2). To this point we have achieved this objective under the extra assumption (A3). If  $P$  is invertible, then  $X = P^{-1}$  meets condition (A3). Also, if  $A = \text{diag}\{\bar{\alpha}_1, \dots, \bar{\alpha}_n\}$  is a diagonal matrix and rank  $P = k$ ,  $0 < k < n$ , then there exists a matrix  $X$  that meets assumption (A3): In this setting there exists a permutation matrix  $\Pi$  such that

$$P = \Pi^* \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix} \Pi,$$

where  $Q_{11}$  is a  $k \times k$  positive definite matrix and

$$Q_{22} = Q_{21}Q_{11}^{-1}Q_{12}.$$

Then, since  $\Pi^*A\Pi$  is still a diagonal matrix, it is easily checked that

$$X = \Pi^* \begin{bmatrix} Q_{11}^{-1} & 0 \\ 0 & 0 \end{bmatrix} \Pi$$

meets assumption (A3). Notice that  $X$  is not a Moore-Penrose inverse unless  $Q_{22} = 0$ . However, if  $A$  is not a diagonal matrix, then there may not be a matrix  $X$  that satisfies assumption (A3). (The problem is Item 2 in (A3), or equivalently the invariance; see Lemma 4.1.) Thus, for example, if  $p = 1$ ,  $q = 2$ ,  $\alpha \in \mathbb{C}_-$ ,  $c \in \mathbb{R}$  and  $c^2 = (\bar{\alpha} - \alpha)/2\pi i$ , then it is readily seen that the three matrices

$$C = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & c \end{bmatrix}, \quad A = \begin{bmatrix} \bar{\alpha} & 1 \\ 0 & \bar{\alpha} \end{bmatrix} \quad \text{and} \quad P = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix},$$

satisfy assumptions (A1)-(A2), since

$$A^*P - PA = \text{diag}\{0, \alpha - \bar{\alpha}\},$$

and

$$2\pi i C^*JC = 2\pi i \text{diag}\{0, -c^2\}.$$

However, a pseudoinverse  $X \geq 0$  must be of the form

$$X = \begin{bmatrix} \bar{\beta} \\ 1 \end{bmatrix} [\beta \quad 1]$$

for some choice of  $\beta \in \mathbb{C}$  and there is no way to choose  $\beta$  so that  $\mathcal{R}_{AX} \subseteq \mathcal{R}_X$ .

The obstacles to invariance may be overcome by modifying the offdiagonal terms of  $A$  “a little” in order to obtain a new upper triangular matrix  $A_0$  such that for this new set of data all three assumptions (A1)-(A3) will hold and

$$\widehat{\mathcal{S}}(C, A, P) = \widehat{\mathcal{S}}(C, A_0, P).$$

The verification of this statement depends upon the analysis of an analogous issue in [BD]. In particular, Theorem 5.3 of [BD] implies the following result:

**THEOREM 8.1.** *If assumptions (A1) and (A2) are in force, and if  $A$  is upper triangular and  $\text{rank } P = k$ , then there exist a pair of upper triangular matrices  $A_0 \in \mathbb{C}^{n \times n}$ ,  $A_3 \in \mathbb{C}^{k \times k}$  and a matrix  $Q \in \mathbb{C}^{n \times k}$  such that:*

- (a)  $PA = PA_0$ .
- (b)  $Q^*PQ > 0$ .
- (c)  $A_0Q = QA_3$ .
- (d)  $\sigma(A_3) \subseteq \sigma(A_0) = \sigma(A)$ .
- (e)  $\widehat{\mathcal{S}}(C, A, P) = \widehat{\mathcal{S}}(C, A_0, P)$ .

**THEOREM 8.2.** *If assumptions (A1)-(A2) are in force for  $(C, A, P)$  and  $A$  is upper triangular, then, in the terminology of Theorem 8.1, assumptions (A1)-(A3) are in force for  $(C, A_0, P)$ .*

**PROOF.** In view of Theorem 8.1, it is readily seen that  $\sigma(A_0) \cap \mathbb{R} = \phi$  and that

$$(8.1) \quad A_0^*P - PA_0 = 2\pi i C^*JC.$$

Let

$$E = Q^*PQ \text{ and } X = QE^{-1}Q^*.$$

Then  $X \geq 0$  and it is easily checked that

$$\text{rank } X = \text{rank } P = k$$

and

$$XPX = QE^{-1}Q^*PQE^{-1}Q^* = X.$$

Thus,

$$\mathbb{C}^n = \mathcal{R}_X + \mathcal{N}_P$$

and hence, upon writing an arbitrary vector  $u \in \mathbb{C}^n$  as

$$u = Xy + v$$

with  $v \in \mathcal{N}_P$ , we see that

$$\begin{aligned} PXPu &= PXP(Xy + v) \\ &= PXPXy \\ &= PXy \\ &= P(Xy + v) = Pu. \end{aligned}$$

This establishes the identity<sup>3</sup>

$$XPX = P .$$

Next, upon multiplying both sides of equation (8.1) by  $X$ , we obtain

$$XA_0^*PX - XPA_0X = 2\pi i XC^*JCX .$$

However, since  $X$  is Hermitian and

$$\begin{aligned} XPA_0X &= XPQA_3E^{-1}Q^* \\ &= QE^{-1}Q^*PQ A_3 E^{-1} Q^* \\ &= QA_3 E^{-1} Q^* \\ &= A_0X , \end{aligned}$$

the last equation exhibits  $X$  as a solution of the Riccati equation

$$(8.2) \quad XA_0^* - A_0X = 2\pi i XC^*JCX .$$

□

## References

- [AD] D. Alpay and H. Dym, On a new class of structured reproducing kernel spaces, *J. Funct. Anal.*, **111** (1993), 1-28.
- [BGR] J.A. Ball, I. Gohberg and L. Rodman, *Interpolation of Rational Matrix Functions*, Birkhäuser, Basel, 1990.
- [BaH] J.A. Ball and J.W. Helton, Interpolation problems of Pick-Nevanlinna and Loewner types for meromorphic matrix functions, *Integral Equations Operator Theory*, **9** (1986), 155-203.
- [Bh] R. Bhatia, *Matrix Analysis*, Springer, New York, 1997.
- [BD] V. Bolotnikov and H. Dym, On degenerate interpolation, entropy and extremal problems for matrix Schur functions, *Integral Equations Operator Theory* **32** (1998), 367-435.
- [Br] L. de Branges, Some Hilbert spaces of analytic functions I, *Trans. Amer. Math. Soc.*, **106** (1963), 445-668.
- [BrR] L. de Branges and J. Rovnyak, Canonical models in quantum scattering theory, in: *Perturbation Theory and its Applications in Quantum Mechanics* (C. Wilcox, ed.) Wiley, New York, 1966, pp. 295-392.
- [Du] V.K. Dubovoj, Indefinite metric in the Schur interpolation problem for analytic matrix functions IV, *Theor. Funktsii, Func. Anal. i Prilozhen.*, **42** (1984), 46-57.
- [D1] H. Dym, *J Contractive Matrices, Reproducing Kernel Hilbert Spaces and Interpolation*, CBMS Regional Conference Series, American Mathematical Society, **71** Providence, R.I., 1989.
- [D2] H. Dym, On reproducing kernel spaces, *J* unitary matrix functions, interpolation and displacement rank, in: *The Gohberg Anniversary Collection*, **OT41**, Birkhäuser (1989), 173-239.
- [D3] H. Dym, Shifts, realizations and interpolation, redux, in: *Oper. Theory: Adv. Appl.*, **OT73** (1994), 182-243.

---

<sup>3</sup>Really this argument shows that

$XPX = X$  and  $\text{rank } X = \text{rank } P \iff XXPX = X$  and  $PXP = P$ .

- [D4] H. Dym, A Basic Interpolation Problem, in: *Holomorphic Spaces* (S. Axler, J. McCarthy and D. Sarason, eds.), Cambridge University Press, Cambridge, England, 1998, pp. 381–423.
- [D5] H. Dym, On Riccati equations and reproducing kernel spaces, in: *Recent Advances in Operator Theory, Oper. Theory: Adv. Appl.*, **OT124**, Birkhäuser, Basel, 2001, pp. 189–215.
- [D6] H. Dym, Reproducing kernels and Riccati equations, *Int. J. Appl. Math. Comp. Science*, **11** (2001), 35–53.
- [D7] H. Dym, Structured matrices, reproducing kernels and interpolation, in: *Structured Matrices in Mathematics, Computer Science and Engineering I* Contemporary Mathematics (V. Olshevsky, ed.), **Vol 280** Amer. Math. Soc., Providence, R.I., 2001, pp. 3–29.
- [KKY] V.E. Katsnelson, A.Ya. Kheifets and P.M. Yuditskii, An abstract interpolation problem and the extension theory of isometric operators, in: *Topics in Interpolation Theory, Oper. Theory Adv. Appl.*, **95**, Birkhäuser, Basel, 1997, pp. 283–298.
- [Kh] A.Ya. Kheifets, Abstract interpolation scheme for harmonic functions, in: *Interpolation Theory, Systems Theory and Related Topics, Oper. Theory Adv. Appl.*, **134**, Birkhäuser, Basel, 2002, pp. 287–317.
- [RR] M. Rosenblum and J. Rovnyak, *Topics in Hardy Classes and Univalent Functions*, Birkhäuser, Basel, 1994.
- [Ro] J. Rovnyak, Characterizations of spaces  $\mathbf{K}(M)$ , unpublished manuscript, 1968.

DEPARTMENT OF MATHEMATICS, THE WEIZMANN INSTITUTE OF SCIENCE, REHOVOT 76100,  
ISRAEL

*E-mail address:* `dym@wisdom.weizmann.ac.il`

*This page intentionally left blank*

## Functions with Pick matrices having bounded number of negative eigenvalues

V. Bolotnikov, A. Kheifets, L. Rodman

**ABSTRACT.** A class is studied of complex valued functions defined on the unit disk (with a possible exception of a discrete set) with the property that all their Pick matrices have not more than a prescribed number of negative eigenvalues. Functions in this class, known to appear as pseudomultipliers of the Hardy space, are characterized in several other ways. It turns out that a typical function in the class is meromorphic with a possible modification at a finite number of points, and total number of poles and of points of modification does not exceed the prescribed number of negative eigenvalues. Bounds are given for the number of points that generate Pick matrices that are needed to achieve the requisite number of negative eigenvalues. A result analogous to Hindmarsh's theorem is proved for functions in the class.

### 1. Introduction and main results

A complex valued function  $S$  is called a *Schur function* if  $S$  is defined on the open unit disk  $\mathbb{D}$ , is analytic, and satisfies  $|S(z)| \leq 1$  for every  $z \in \mathbb{D}$ . Schur functions and their generalizations have been extensively studied in the literature. They admit various useful characterizations; one such well-known characterization is the following: *A function  $S$  defined on  $\mathbb{D}$  is a Schur function if and only the kernel*

$$(1.1) \quad K_S(z, w) = \frac{1 - S(z)S(w)^*}{1 - zw^*}$$

*is positive on  $\mathbb{D}$ .* The positivity of (1.1) on  $\mathbb{D}$  means that for every choice of positive integer  $n$  and of distinct points  $z_1, \dots, z_n \in \mathbb{D}$ , the *Pick matrix*

$$P_n(S; z_1, \dots, z_n) = \left[ \frac{1 - S(z_i)S(z_j)^*}{1 - z_i z_j^*} \right]_{i,j=1}^n$$

is positive semidefinite:  $P_n(S; z_1, \dots, z_n) \geq 0$ .

---

1991 *Mathematics Subject Classification.* 30E99, 15A63.

*Key words and phrases.* Schur functions, Pick matrices.

The research of the third author is partially supported by an NSF Grant.

The following remarkable theorem due to Hindmarsh [12] (see also [10]) implies that if all Pick matrices of order 3 are positive semidefinite, then the function is necessarily analytic.

**Theorem 1.1.** *Let  $S$  be a function defined on an open set  $U \subseteq \mathbb{D}$ , and assume that  $P_3(S; z_1, z_2, z_3)$  is positive semidefinite for every choice of  $z_1, z_2, z_3 \in U$ . Then  $S$  is analytic on  $U$ , and  $|S(z)| \leq 1$  for every  $z \in U$ .*

A corollary of Theorem 1.1 will be useful; we say that a set  $\Lambda \subset \mathbb{D}$  is *discrete* (in  $\mathbb{D}$ ) if  $\Lambda$  is at most countable, with accumulation points (if any) only on the boundary of  $\mathbb{D}$ .

**Corollary 1.2.** *Let  $S$  be a function defined on  $\mathbb{D} \setminus \Lambda$ , where  $\Lambda$  is a discrete set. If  $P_3(S; z_1, z_2, z_3)$  is positive semidefinite for every choice of  $z_1, z_2, z_3 \in \mathbb{D} \setminus \Lambda$ , then  $S$  admits analytic continuation to a Schur function (defined on  $\mathbb{D}$ ).*

In this paper we prove, among other things, an analogue of Hindmarsh's theorem for the class of functions whose Pick matrices have a bounded number of negative eigenvalues. These functions need not be analytic, or meromorphic, on  $\mathbb{D}$ . It will be generally assumed that such functions are defined on  $\mathbb{D} \setminus \Lambda$ , where  $\Lambda$  is a discrete set which may depend on the function. More precisely:

**Definition 1.3.** *Given a nonnegative integer  $\kappa$ , the class  $\mathcal{S}_\kappa$  consists of (complex valued) functions  $f$  defined on  $\text{Dom}(f) = \mathbb{D} \setminus \Lambda$ , where  $\Lambda = \Lambda(f)$  is a discrete set, and such that all Pick matrices*

$$(1.2) \quad P_n(f; z_1, \dots, z_n) := \left[ \frac{1 - f(z_i)f(z_j)^*}{1 - z_i z_j^*} \right]_{i,j=1}^n, \quad z_1, \dots, z_n \in \mathbb{D} \setminus \Lambda \text{ are distinct,}$$

*have at most  $\kappa$  negative eigenvalues, and at least one such Pick matrix has exactly  $\kappa$  negative eigenvalues (counted with multiplicities).*

The Pick matrices (1.2) are clearly Hermitian. Note also that in Definition 1.3 the points  $z_1, \dots, z_n \in \mathbb{D} \setminus \Lambda$  are assumed to be distinct; an equivalent definition is obtained if we omit the requirement that the points  $z_1, \dots, z_n$  are distinct.

Meromorphic functions in the class  $\mathcal{S}_\kappa$  have been studied before in various contexts: approximation problems [3], spectral theory of unitary operators in Pontryagin spaces [13], Schur–Takagi problem [1], Nevanlinna–Pick problem [15], [11], [6], model theory [4]. Recently, functions with jumps in  $\mathcal{S}_\kappa$  appeared in the theory of almost multipliers (or pseudomultipliers) [2]; this connection will be discussed in more details later on.

Throughout the paper,  $\text{Dom}(f)$  stands for the domain of definition of  $f$  and  $Z(f)$  denotes the zero set for  $f$ :

$$Z(f) = \{z \in \text{Dom}(f) : f(z) = 0\}.$$

The number of negative eigenvalues (counted with multiplicities) of a Hermitian matrix  $P$  will be denoted by  $\text{sq}_-P$ . For a function  $f$  defined on  $\mathbb{D} \setminus \Lambda$ , where  $\Lambda$  is a discrete set, we let  $\mathbf{k}_n(f)$  denote the maximal number of negative eigenvalues (counted with multiplicities) of all Pick matrices of order  $n$ :

$$(1.3) \quad \mathbf{k}_n(f) := \max_{z_1, \dots, z_n \in \mathbb{D} \setminus \Lambda} \text{sq}_-P_n(f; z_1, \dots, z_n).$$

We also put formally  $\mathbf{k}_0(f) = 0$ . It is clear that the sequence  $\{\mathbf{k}_n(f)\}$  is non decreasing and if  $f \in \mathcal{S}_\kappa$ , then

$$(1.4) \quad \mathbf{k}_n(f) = \kappa$$

for all sufficiently large integers  $n$ . Note that the class  $\mathcal{S}_0$  coincides with the Schur class; more precisely, every function  $f$  in  $\mathcal{S}_0$  admits an extension to a Schur function. Here and elsewhere, we say that a function  $g$  is an *extension* of a function  $f$  if  $\text{Dom}(g) \supseteq \text{Dom}(f)$  and  $g(z) = f(z)$  for every  $z \in \text{Dom}(f)$ .

The following result by Krein - Langer [13] gives a characterization of *meromorphic* functions in  $\mathcal{S}_\kappa$ .

**Theorem 1.4.** *Let  $f$  be a meromorphic function on  $\mathbb{D}$ . Then  $f \in \mathcal{S}_\kappa$  if and only if  $f(z)$  can be represented as  $f(z) = \frac{S(z)}{B(z)}$ , where  $S$  is a Schur function and  $B$  is a Blaschke product of degree  $\kappa$  such that  $S$  and  $B$  have no common zeros.*

Recall that a *Blaschke product* (all Blaschke products in this paper are assumed to be finite) is a rational function  $B(z)$  that is analytic on  $\mathbb{D}$  and unimodular on the unit circle  $\mathbb{T} : |B(z)| = 1$  for  $|z| = 1$ ; the *degree* of  $B(z)$  is the number of zeros (counted with multiplicities) of  $B(z)$  in  $\mathbb{D}$ . See also [11], [9], [8] for various proofs of matrix and operator-valued versions of Theorem 1.4.

However, not all functions in  $\mathcal{S}_\kappa$  are meromorphic, as a standard example shows:

**Example 1.5.** Let the function  $f$  be defined on  $\mathbb{D}$  as  $f(z) = 1$  if  $z \neq 0$ ;  $f(0) = 0$ . Then  $P_n(f; z_1, \dots, z_n) = 0$  if  $z_j$ 's are all nonzero; if one of the points is zero (up to a permutation similarity we can assume that  $z_1 = 0$ ), then  $P_n(f; z_1, \dots, z_n)$  is the matrix with ones in the first column and in the first row and with zeroes elsewhere. This matrix clearly is of rank two and has one negative eigenvalue. Thus,  $f \in \mathcal{S}_1$ . That  $f$  has a jump discontinuity at  $z = 0$  is essential; removing the discontinuity would result in the constant function  $\tilde{f}(z) = 1$ , which does not belong to  $\mathcal{S}_1$ .

As it was shown in [2], functions in  $\mathcal{S}_\kappa$  that are defined in  $\mathbb{D}$  except for a finite set of singularities, are exactly the  $\kappa$ -pseudomultipliers of the Hardy space  $H_2$  (see [2] for the definitions of pseudomultipliers and their singularities; roughly speaking, a pseudomultiplier maps a subspace of finite codimension in a Hilbert space of functions into the same Hilbert space, by means of the corresponding multiplication operator). In fact, similar results were obtained in [2] for more general reproducing kernel Hilbert spaces of functions. Theorem 2.1 of [2] implies in particular that every function in the class  $\mathcal{S}_\kappa$  defined on a set of uniqueness of  $H_2$  can be extended to a function in  $\mathcal{S}_\kappa$  defined everywhere in  $\mathbb{D}$  except for a finite number of singularities. Theorem 3.1 of [2] implies that the pseudomultipliers of  $H_2$  are meromorphic in  $\mathbb{D}$  with a finite number of poles and jumps, some of which may coincide. This result applies to other reproducing kernel Hilbert spaces of analytic functions as well, as proved in [2].

For the Hardy space  $H_2$ , the pseudomultipliers can be characterized in a more concrete way, namely as standard functions defined below, see Theorem 1.7(2). This characterization is based largely on Krein - Langer theorem 1.4.

In this paper we focus on a more detailed study of the class  $\mathcal{S}_\kappa$ . Our proofs depend on techniques used in interpolation, such as matrix inequalities and Schur

complements, and allow us to obtain generalizations of Hindmarsh's theorem (Theorem 1.7(3)) and precise estimates of the size of Pick matrices needed to attain the required number of negative eigenvalues (Theorem 1.8).

**Definition 1.6.** *A function  $f$  is said to be a standard function if it admits the representation*

$$(1.5) \quad f(z) = \begin{cases} \frac{S(z)}{B(z)} & \text{if } z \notin \mathcal{W} \cup \mathcal{Z}, \\ \gamma_j & \text{if } z = z_j \in \mathcal{Z}, \end{cases}$$

for some complex numbers  $\gamma_1, \dots, \gamma_\ell$ , where:

- (1)  $\mathcal{Z} = \{z_1, \dots, z_\ell\}$  and  $\mathcal{W} = \{w_1, \dots, w_p\}$  are disjoint sets of distinct points in  $\mathbb{D}$ ;
- (2)  $B(z)$  is a Blaschke product of degree  $q \geq 0$  and  $S(z)$  is a Schur function with the zero sets  $Z(B)$  and  $Z(S)$ , respectively, such that

$$\mathcal{W} \subseteq Z(B) \subseteq \mathcal{W} \cup \mathcal{Z} \quad \text{and} \quad Z(B) \cap Z(S) = \emptyset;$$

- (3) if  $z_j \in \mathcal{Z} \setminus Z(B)$ , then  $\frac{S(z_j)}{B(z_j)} \neq \gamma_j$ .

For the standard function  $f$  of the form (1.5),  $\text{Dom}(f) = \mathbb{D} \setminus \mathcal{W}$ . More informally,  $\mathcal{Z}$  is the set of points  $z_0$  at which  $f$  is defined and  $f(z_0) \neq \lim_{z \rightarrow z_0} f(z)$ . The case when  $\lim_{z \rightarrow z_0} |f(z)| = \infty$  is not excluded here; in this case  $f(z_0)$  is defined nevertheless. The set  $\mathcal{W}$  consists of those poles of  $\frac{S(z)}{B(z)}$  at which  $f$  is not defined.

In reference to the properties (1)-(3) in Definition 1.6 we will say that  $f(z)$  has  $q$  poles (the zeros of  $B(z)$ ), where each pole is counted according to its multiplicity as a zero of  $B(z)$ , and  $\ell$  jumps  $z_1, \dots, z_\ell$ . Note that the poles and jumps need not be disjoint.

**Theorem 1.7.** *Let  $f$  be defined on  $\mathbb{D} \setminus \Lambda$ , where  $\Lambda$  is a discrete set. Fix a nonnegative integer  $\kappa$ . Then the following statements are equivalent:*

- (1)  $f$  belongs to  $\mathcal{S}_\kappa$ .
- (2)  $f$  admits an extension to a standard function with  $\ell$  jumps (for some  $\ell$ ,  $0 \leq \ell \leq \kappa$ ) and  $\kappa - \ell$  poles, and the jumps of the standard function are contained in  $\mathbb{D} \setminus \Lambda$ .
- (3)

$$(1.6) \quad \mathbf{k}_n(f) = \mathbf{k}_{n+3}(f) = \kappa \quad \text{for some integer } n \geq 0.$$

Note that the extension to a standard function in the second statement is unique. Note also that the equivalence **1**  $\Leftrightarrow$  **3** is a generalization of Hindmarsh's theorem to the class of functions whose Pick matrices have a bounded number of negative eigenvalues, and coincides with Theorem 1.1 if  $\kappa = 0$ .

The third statement in Theorem 1.7 raises the question of the minimal possible value of  $n$  for which (1.4) holds. For convenience of future reference we denote this minimal value by  $N(f)$ :

$$(1.7) \quad N(f) = \min_n \{n : \mathbf{k}_n(f) = \kappa\}, \quad f \in \mathcal{S}_\kappa.$$

**Theorem 1.8.** *Let  $f$  be a standard function with  $q$  poles and  $\ell$  jumps. Then  $f \in \mathcal{S}_\kappa$ , where  $\kappa = q + \ell$ , and*

$$(1.8) \quad q + \ell \leq N(f) \leq q + 2\ell.$$

The left inequality in (1.8) is self-evident (to have  $\kappa$  negative eigenvalues, a Hermitian matrix must be of size at least  $\kappa \times \kappa$ ), while the right inequality is the essence of the theorem. We shall show that inequalities (1.8) are exact; here we present a simple example showing that these inequalities can be strict.

**Example 1.9.** Let

$$f(z) = \begin{cases} 1, & z \neq 0, \\ a \neq 1, & z = 0. \end{cases}$$

Then  $f(z)$  is a standard function from  $\mathcal{S}_1$  with no poles and one jump and Theorem 1.8 guarantees that  $1 \leq N(f) \leq 2$ . More detailed analysis shows that if  $|a| > 1$ , then  $\mathbf{k}_n(f) = 1$  for  $n \geq 1$  and therefore  $1 = N(f) < 2$ . If  $|a| \leq 1$ , then  $\mathbf{k}_1(f) = 0$ ,  $\mathbf{k}_n(f) = 1$  for  $n \geq 2$  and therefore  $1 < N(f) = 2$ .

The following addition to Theorem 1.8 is useful:

**Remark 1.10.** *Let  $f$  be as in Theorem 1.8. Let  $w_1, \dots, w_k$  be the distinct poles of  $f$  with multiplicities  $r_1, \dots, r_k$ , respectively, and let  $z_1, \dots, z_\ell$  be the (distinct) jumps of  $f$ . Then there exists an  $\epsilon > 0$  such that every set  $Y = \{y_1, \dots, y_m\}$  of  $m := q + 2\ell$  distinct points satisfying the conditions 1 - 3 listed below has the property that*

$$\text{sq\_}P_m(f; y_1, \dots, y_m) = q + \ell.$$

- (1)  $\ell$  points in the set  $Y$  coincide with  $z_1, \dots, z_\ell$ ;
- (2)  $\ell$  points in  $Y$ , different from  $z_1, \dots, z_\ell$ , are at a distance less than  $\epsilon$  from the set  $\{z_1, \dots, z_\ell\}$ ;
- (3) the remaining  $q$  points in  $Y$  are distributed over  $k$  disjoint subsets  $Y_1, \dots, Y_k$  such that the set  $Y_j$  consists of  $r_j$  points all at a positive distance less than  $\epsilon$  from  $w_j$ , for  $j = 1, \dots, k$ .

The proof of Remark 1.10 is obtained in the course of the proof of Theorem 1.8.

Still another characterization of the class  $\mathcal{S}_\kappa$  is given in the following theorem:

**Theorem 1.11.** *Let  $f$  be as in Theorem 1.7, and fix a nonnegative integer  $\kappa$ . Then:*

- (1) *If  $\mathbf{k}_{2\kappa}(f) = \mathbf{k}_{2\kappa+3}(f) = \kappa$ , then  $f \in \mathcal{S}_\kappa$ .*
- (2) *If  $f \in \mathcal{S}_\kappa$ , then  $\mathbf{k}_{2\kappa}(f) = \kappa$ .*

Example 1.9 (with  $|a| \leq 1$ ) shows that for  $\kappa = 1$ , the subscript  $2\kappa$  in Theorem 1.11 cannot be replaced by any smaller nonnegative integer.

Theorems 1.7, 1.8, and 1.11 comprise the main results of this paper.

Sections 2 through 4 contain the proofs of Theorems 1.7 and 1.8. In Section 5, a local result is proved to the effect that for a function  $f \in \mathcal{S}_\kappa$  and any open set  $\Omega$  in  $\text{Dom}(f)$ , the requisite number of negative eigenvalues of Pick matrices  $P_n(f; z_1, \dots, z_n)$  can be achieved when the points  $z_j$  are restricted to belong to  $\Omega$ .

Hindmarsh sets and their elementary properties are introduced in Section 6, where we also state an open problem.

If not stated explicitly otherwise, all functions are assumed to be scalar (complex valued). The superscript \* stands for the conjugate of a complex number or the conjugate transpose of a matrix.

## 2. Theorem 1.7: part of the proof

In this section we prove implication  $3 \Rightarrow 1$  of Theorem 1.7. We start with some auxiliary lemmas.

**Lemma 2.1.** *If an  $m \times m$  matrix  $X$  has rank  $k < m$  and the algebraic multiplicity of zero as an eigenvalue of  $X$  is  $m - k$ , then there exists a  $k \times k$  nonsingular principal submatrix of  $X$ .*

**Proof:** The characteristic polynomial of  $X$  has the form  $\lambda^m + a_{m-1}\lambda^{m-1} + \dots + a_{m-k}\lambda^{m-k}$ , where  $a_{m-k} \neq 0$ . Since  $\pm a_{m-k}$  is the sum of all determinants of  $k \times k$  principal submatrices of  $X$ , at least one of those determinants must be nonzero.  $\square$

**Lemma 2.2.** *Let  $X$  be a Hermitian  $n \times n$  matrix. Then*

- (1)  $\text{sq\_}_Y \geq \text{sq\_}_X$  for all Hermitian matrices  $Y \in \mathbb{C}^{n \times n}$  in some small neighborhood of  $X$ .
- (2)  $\text{sq\_}_P^*XP \leq \text{sq\_}_X$  for every  $P \in \mathbb{C}^{n \times m}$ . In particular, if  $X_0$  is any principal submatrix of  $X$ , then  $\text{sq\_}_X_0 \leq \text{sq\_}_X$ .
- (3) If  $\{X_m\}_{m=1}^\infty$  is a sequence of Hermitian matrices such that  $\text{sq\_}_X_m \leq k$  for  $m = 1, 2, \dots$ , and  $\lim_{m \rightarrow \infty} X_m = X$ , then also  $\text{sq\_}_X \leq k$ .

**Proof:** The first and third statements easily follow by the continuity properties of eigenvalues of Hermitian matrices. The second statement is a consequence from the interlacing properties of eigenvalues of Hermitian matrices, see, e.g. [14, Section 8.4].  $\square$

**Lemma 2.3.** *Let  $f$  be a function defined on  $\mathbb{D} \setminus \Lambda$ , where  $\Lambda$  is a discrete set.*

- (1) *If  $g$  is defined on a set  $(\mathbb{D} \setminus \Lambda) \cup \{w_1, \dots, w_k\}$  (the points  $w_1, \dots, w_k \in \mathbb{D}$  are not assumed to be distinct or to be disjoint with  $\Lambda$ ) and  $g(z) = f(z)$  for every  $z \in \mathbb{D} \setminus \Lambda$ , then*
- (2.1)  $\mathbf{k}_n(f) \leq \mathbf{k}_n(g) \leq \max_{0 \leq r \leq \min\{k, n\}} \{\mathbf{k}_{n-r}(f) + r\} \leq \mathbf{k}_n(f) + k, \quad n = 1, 2, \dots$
- (2) *If  $g$  is defined on  $\mathbb{D} \setminus \Lambda$ , and  $g(z) = f(z)$  for every  $z \in \mathbb{D} \setminus \Lambda$ , except for  $k$  distinct points  $z_1, \dots, z_k \in \mathbb{D} \setminus \Lambda$ , then*

$$\mathbf{k}_n(g) \leq \mathbf{k}_n(f) + k, \quad n = 1, 2, \dots$$

**Proof:** We prove the first statement. The first inequality in (2.1) follows from the definition (1.3) of  $\mathbf{k}_n(f)$ . Let  $y_1, \dots, y_n$  be a set of distinct points in  $(\mathbb{D} \setminus \Lambda) \cup \{w_1, \dots, w_k\}$ , and assume that exactly  $r$  points among  $y_1, \dots, y_n$  are in the set  $\{w_1, \dots, w_k\} \setminus (\mathbb{D} \setminus \Lambda)$ . For notational convenience, suppose that  $y_1, \dots, y_r \in \{w_1, \dots, w_k\} \setminus (\mathbb{D} \setminus \Lambda)$ . We have

$$P_n(g; y_1, \dots, y_n) = \begin{bmatrix} * & * \\ * & P_{n-r}(f; y_{r+1}, \dots, y_n) \end{bmatrix},$$

where  $*$  denotes blocks of no immediate interest. By the interlacing properties of eigenvalues of Hermitian matrices, we have

$$\text{sq}_- P_n(g; y_1, \dots, y_n) \leq \text{sq}_- P_{n-r}(f; y_{r+1}, \dots, y_n) + r \leq \mathbf{k}_{n-r}(f) + r,$$

and, since  $0 \leq r \leq \min\{n, k\}$ , the second and third inequalities in (2.1) follow.

For the second statement, using induction on  $k$ , we need to prove only the case  $k = 1$ . Let  $y_1, \dots, y_n$  be a set of distinct points in  $\mathbb{D} \setminus \Lambda$ . Then

$$(2.2) \quad P_n(g; y_1, \dots, y_n) = P_n(f; y_1, \dots, y_n) + Q,$$

where  $Q$  is either the zero matrix (if  $y_j \neq z_1$ ), or  $Q$  has an  $(n-1) \times (n-1)$  principal zero submatrix (if  $y_j = z_1$  for some  $j$ ), and in any case  $Q$  has at most one negative eigenvalue. Let  $k = \text{sq}_- P_n(g; y_1, \dots, y_n)$ . Then (2.2) implies, using the Weyl inequalities for eigenvalues of the sum of two Hermitian matrices (see, e.g., [7, Section III.2]) that  $\text{sq}_- P_n(f; y_1, \dots, y_n) \leq k + 1$ .  $\square$

**Proof of 3  $\Rightarrow$  1 of Theorem 1.7.** Let  $f$  be defined on  $\mathbb{D} \setminus \Lambda$ , where  $\Lambda$  is discrete. Furthermore, let (1.6) hold for some integer  $n \geq 0$  and let the set

$$\mathcal{Z} = \{z_1, \dots, z_n\} \subset \mathbb{D} \setminus \Lambda$$

be such that

$$\text{sq}_- P_n(f; z_1, \dots, z_n) = \kappa.$$

Without loss of generality we can assume that

$$(2.3) \quad P_n(f; z_1, \dots, z_n) = \left[ \frac{1 - f(z_i)f(z_j)^*}{1 - z_i z_j^*} \right]_{i,j=1}^n$$

is not singular. Indeed, if it happens that the matrix (2.3) is singular, and its rank is  $m < n$ , then by Lemma 2.1, there is a nonsingular  $m \times m$  principal submatrix  $P_m(f; z_{i_1}, \dots, z_{i_m})$  of (2.3). A Schur complement argument shows that

$$\text{sq}_- P_m(f; z_{i_1}, \dots, z_{i_m}) = \kappa.$$

It follows then that  $\mathbf{k}_m(f) = \kappa$ . But then, in view of (1.6) and the nondecreasing property of the sequence  $\{\mathbf{k}_j(f)\}_{j=1}^\infty$ , we have  $\mathbf{k}_{m+3}(f) = \kappa$ , and the subsequent proof may be repeated with  $n$  replaced by  $m$ .

Setting  $P_n(f; z_1, \dots, z_n) = P$  for short, note the identity

$$(2.4) \quad P - TPT^* = FJF^*,$$

where

$$(2.5) \quad T = \begin{bmatrix} z_1 & & \\ & \ddots & \\ & & z_n \end{bmatrix}, \quad J = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad F = \begin{bmatrix} 1 & f(z_1) \\ \vdots & \vdots \\ 1 & f(z_n) \end{bmatrix}.$$

Consider the function

$$(2.6) \quad \Theta(z) = I_2 - (1 - z)F^*(I_n - zT^*)^{-1}P^{-1}(I_n - T)^{-1}F$$

It follows from (2.4) by a straightforward standard calculation (see, e.g., [5, Section 7.1]) that

$$(2.7) \quad J - \Theta(z)J\Theta(w)^* = (1 - zw^*)F^*(I_n - zT^*)^{-1}P^{-1}(I_n - w^*T)^{-1}F.$$

Note that  $\Theta(z)$  is a rational function taking  $J$ -unitary values on  $\mathbb{T}$ :  $\Theta(z)J\Theta(z)^* = J$  for  $z \in \mathbb{T}$ . Therefore, by the symmetry principle,

$$\Theta(z)^{-1} = J\Theta(\frac{1}{z^*})^*J = I_2 + (1-z)F^*(I_n - T^*)^{-1}P^{-1}(zI_n - T)^{-1}FJ,$$

which implies, in particular, that  $\Theta(z)$  is invertible at each point  $z \notin \mathcal{Z}$ . By (1.6) and by Lemma 2.2,

$$(2.8) \quad \text{sq}_- P_{n+3}(f; z_1, \dots, z_n, \zeta_1, \zeta_2, \zeta_3) = \kappa$$

for every choice of  $\zeta_1, \zeta_2, \zeta_3 \in \mathbb{D} \setminus \Lambda$ . The matrix in (2.8) can be written in the block form as

$$(2.9) \quad P_{n+3}(f; z_1, \dots, z_n, \zeta_1, \zeta_2, \zeta_3) = \begin{bmatrix} P & \Psi^* \\ \Psi & P_3(f; \zeta_1, \zeta_2, \zeta_3) \end{bmatrix},$$

where

$$\Psi = \begin{bmatrix} \Psi_1 \\ \Psi_2 \\ \Psi_3 \end{bmatrix} \quad \text{and} \quad \Psi_i = \begin{bmatrix} \frac{1-f(\zeta_i)f(z_1)^*}{1-\zeta_iz_1^*} & \dots & \frac{1-f(\zeta_i)f(z_n)^*}{1-\zeta_iz_n^*} \end{bmatrix} \quad (i = 1, 2, 3).$$

The last formula for  $\Psi_i$  can be written in terms of (2.5) as

$$(2.10) \quad \Psi_i = [1 - f(\zeta_i)]F^*(I_n - \zeta_i T^*)^{-1} \quad (i = 1, 2, 3).$$

By (1.6), it follows from (2.9) that

$$P_3(f; \zeta_1, \zeta_2, \zeta_3) - \Psi P^{-1} \Psi^* \geq 0,$$

or more explicitly,

$$(2.11) \quad \left[ \frac{1-f(\zeta_i)f(\zeta_j)^*}{1-\zeta_i\zeta_j^*} - \Psi_i P^{-1} \Psi_j^* \right]_{i,j=1}^3 \geq 0.$$

By (2.10) and (2.7),

$$\begin{aligned} & \frac{1-f(\zeta_i)f(\zeta_j)^*}{1-\zeta_i\zeta_j^*} - \Psi_i P^{-1} \Psi_j^* \\ &= [1 - f(\zeta_i)] \left\{ \frac{J}{1-\zeta_i\zeta_j^*} - F^*(I_n - \zeta_i T^*)^{-1} P^{-1} (I_n - \zeta_j^* T)^{-1} F \right\} \begin{bmatrix} 1 \\ -f(\zeta_j)^* \end{bmatrix} \\ (2.12) \quad &= [1 - f(\zeta_i)] \frac{\Theta(\zeta_i)J\Theta(\zeta_j)^*}{1-\zeta_i\zeta_j^*} \begin{bmatrix} 1 \\ -f(\zeta_j)^* \end{bmatrix}, \end{aligned}$$

which allows us to rewrite (2.11) as

$$(2.13) \quad \left[ [1 - f(\zeta_i)] \frac{\Theta(\zeta_i)J\Theta(\zeta_j)^*}{1-\zeta_i\zeta_j^*} \begin{bmatrix} 1 \\ -f(\zeta_j)^* \end{bmatrix} \right]_{i,j=1}^3 \geq 0.$$

Introducing the block decomposition

$$\Theta(z) = \begin{bmatrix} \theta_{11}(z) & \theta_{12}(z) \\ \theta_{21}(z) & \theta_{22}(z) \end{bmatrix}$$

of  $\Theta$  into four scalar blocks, note that

$$(2.14) \quad d(z) := \theta_{21}(z)f(z) - \theta_{11}(z) \neq 0, \quad z \in \mathbb{D} \setminus (\mathcal{Z} \cup \Lambda).$$

Indeed, assuming that  $\theta_{21}(\zeta)f(\zeta) = \theta_{11}(\zeta)$  for some  $\zeta \in \mathbb{D} \setminus (\mathcal{Z} \cup \Lambda)$ , we get

$$\begin{aligned} & [1 \ -f(\zeta)] \frac{\Theta(\zeta)J\Theta(\zeta)^*}{1-|\zeta|^2} \begin{bmatrix} 1 \\ -f(\zeta)^* \end{bmatrix} \\ &= -[1 \ -f(\zeta)] \frac{\begin{bmatrix} \theta_{12}(\zeta) \\ \theta_{22}(\zeta) \end{bmatrix} \begin{bmatrix} \theta_{12}(\zeta)^* & \theta_{22}(\zeta)^* \end{bmatrix}}{1-|\zeta|^2} \begin{bmatrix} 1 \\ -f(\zeta)^* \end{bmatrix} \leq 0, \end{aligned}$$

which contradicts (2.13), unless  $\det \Theta(\zeta) = 0$ . But as we have mentioned above,  $\Theta(z)$  is invertible at each point  $\zeta \notin \mathcal{Z}$ .

Thus, the function

$$(2.15) \quad \sigma(z) = \frac{\theta_{12}(z) - f(z)\theta_{22}(z)}{\theta_{21}(z)f(z) - \theta_{11}(z)}$$

is defined on  $\mathbb{D} \setminus (\mathcal{Z} \cup \Lambda)$ . Moreover,

$$[1 \ -f(\zeta)] \Theta(\zeta) = -d(\zeta)^{-1} [1 \ -\sigma(\zeta)]$$

and thus, inequality (2.13) can be written in terms of  $\sigma$  as

$$\left[ d(\zeta_i)^{-1} [1 \ -\sigma(\zeta_i)] \frac{J}{1-\zeta_i\zeta_j^*} \begin{bmatrix} 1 \\ -\sigma(\zeta_j)^* \end{bmatrix} (d(\zeta_j)^*)^{-1} \right]_{i,j=1}^3 \geq 0,$$

or equivalently, (since  $d(\zeta_i) \neq 0$ ) as

$$\left[ \frac{1 - \sigma(\zeta_i)\sigma(\zeta_j)^*}{1 - \zeta_i\zeta_j^*} \right]_{i,j=1}^3 \geq 0.$$

The latter inequality means that  $P_3(\sigma; \zeta_1, \zeta_2, \zeta_3) \geq 0$  for every choice of points  $\zeta_1, \zeta_2, \zeta_3$  in  $\mathbb{D} \setminus (\mathcal{Z} \cup \Lambda)$ . By Corollary 1.2,  $\sigma(z)$  is a Schur function. Although  $\sigma(z)$  has been defined via (2.15) on  $\mathbb{D} \setminus (\mathcal{Z} \cup \{z_1, \dots, z_n\})$ , it admits an analytic continuation to all of  $\mathbb{D}$ , which still will be denoted by  $\sigma(z)$ . It follows from (2.15) that  $f$  coincides with the function

$$(2.16) \quad F(z) = \frac{\theta_{11}(z)\sigma(z) + \theta_{12}(z)}{\theta_{21}(z)\sigma(z) + \theta_{22}(z)}.$$

at every point  $z \in \mathbb{D} \setminus (\mathcal{Z} \cup \Lambda)$ . Since  $f$  has not been defined on  $\Lambda$ , one can consider  $F$  as a (unique) meromorphic extension of  $f$ . However,  $F$  need not coincide with  $f$  on  $\mathcal{Z}$ .

Now we prove that  $f \in \mathcal{S}_\kappa$ . To this end it suffices to show that

$$(2.17) \quad \text{sq}_- P_{n+r}(f; z_1, \dots, z_n, \zeta_1, \dots, \zeta_r) = \kappa$$

for every choice of  $\zeta_1, \dots, \zeta_r \in \mathbb{D} \setminus (\mathcal{Z} \cup \Lambda)$ . Note that all possible ‘‘jumps’’ of  $f$  are in  $\mathcal{Z}$  and at all other points of  $\mathbb{D} \setminus \Lambda$ , it holds that  $f(\zeta) = F(\zeta)$ . Thus, writing

$$P_{n+r}(f; z_1, \dots, z_n, \zeta_1, \dots, \zeta_r) = \begin{bmatrix} P & \Psi^* \\ \Psi & P_r(f; \zeta_1, \dots, \zeta_r) \end{bmatrix},$$

where

$$\Psi = \begin{bmatrix} \Psi_1 \\ \vdots \\ \Psi_r \end{bmatrix}$$

and the  $\Psi_i$ 's are defined via (2.10) for  $i = 1, \dots, r$ , we conclude by the Schur complement argument that

$$(2.18) \quad \text{sq}_- P_{n+r}(f; z_1, \dots, z_n, \zeta_1, \dots, \zeta_r) = \text{sq}_- P + \text{sq}_-(P_r(f; , \zeta_1, \dots, \zeta_r) - \Psi P^{-1} \Psi^*).$$

It follows from the calculation (2.12) that

$$P_r(f; \zeta_1, \dots, \zeta_r) - \Psi P^{-1} \Psi^* = \left[ \begin{bmatrix} 1 & -f(\zeta_i) \end{bmatrix} \frac{\Theta(\zeta_i) J \Theta(\zeta_j)^*}{1 - \zeta_i \zeta_j^*} \begin{bmatrix} 1 \\ -f(\zeta_j)^* \end{bmatrix} \right]_{i,j=1}^r,$$

which can be written in terms of functions  $\sigma$  and  $d$  defined in (2.15) and (2.14), respectively, as

$$P_r(f; \zeta_1, \dots, \zeta_r) - \Psi P^{-1} \Psi^* = \left[ d(\zeta_i)^{-1} \frac{1 - \sigma(\zeta_i) \sigma(\zeta_j)^*}{1 - \zeta_i \zeta_j^*} (d(\zeta_j)^*)^{-1} \right]_{i,j=1}^r.$$

We have already proved that  $\sigma$  is a Schur function and therefore,

$$P_r(f; \zeta_1, \dots, \zeta_r) - \Psi P^{-1} \Psi^* \geq 0,$$

which together with (2.18) implies (2.17).  $\square$

### 3. Proof of Theorem 1.8

Since the first inequality in (1.8) is obvious, we need to prove only the second inequality.

If  $\tilde{f}$  is the meromorphic part of  $f$ , then by Theorem 1.4  $\tilde{f} \in \mathcal{S}_q$ , and therefore by Lemma 2.3,  $\mathbf{k}_n(f) \leq q + \ell$ , for  $n = 1, 2, \dots$ . Therefore, it suffices to prove that there exist  $q + 2\ell$  distinct points  $u_1, \dots, u_{q+2\ell} \in \text{Dom}(f)$  such that

$$\text{sq}_- P_{q+2\ell}(f; u_1, \dots, u_{q+2\ell}) \geq q + \ell.$$

We start with notation and some preliminary results. Let

$$J_r(a) = \begin{bmatrix} a & 0 & 0 & \cdots & 0 \\ 1 & a & 0 & \cdots & 0 \\ 0 & 1 & a & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & a \end{bmatrix}, \quad a \in \mathbb{C}$$

be the lower triangular  $r \times r$  Jordan block with eigenvalue  $a$  and let  $E_r$  and  $G_r$  be vectors from  $\mathbb{C}^r$  defined by

$$(3.1) \quad E_r = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \in \mathbb{C}^r \quad \text{and} \quad G_r = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbb{C}^r.$$

Given an ordered set  $\mathcal{Z} = \{z_1, \dots, z_k\}$  of distinct points in the complex plane, we denote by  $\Phi(\mathcal{Z})$  the lower triangular  $k \times k$  matrix defined by

$$(3.2) \quad \Phi(\mathcal{Z}) = [\Phi_{i,j}]_{i,j=1}^k, \quad \Phi_{i,j} = \begin{cases} \frac{1}{\phi'_i(z_j)} & \text{if } i \geq j, \\ 0 & \text{if } i < j, \end{cases}$$

where  $\phi_i(z) = \prod_{j=1}^i (z - z_j)$ . Furthermore, given a set  $\mathcal{Z}$  as above, for a complex valued function  $v(z)$  we define recursively the divided differences  $[z_1, \dots, z_j]_v$  by

$$[z_1]_v = v(z_1), \quad [z_1, \dots, z_{j+1}]_v = \frac{[z_1, \dots, z_j]_v - [z_2, \dots, z_{j+1}]_v}{z_1 - z_{j+1}}, \quad j = 1, \dots, k,$$

and use the following notation for associated matrices and vectors:

$$D(\mathcal{Z}) = \begin{bmatrix} z_1 & 0 & \cdots & 0 \\ 0 & z_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & z_k \end{bmatrix}, \quad J(\mathcal{Z}) = \begin{bmatrix} z_1 & 0 & 0 & \cdots & 0 \\ 1 & z_2 & 0 & \cdots & 0 \\ 0 & 1 & z_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & z_k \end{bmatrix},$$

$$(3.3) \quad v(\mathcal{Z}) = \begin{bmatrix} v(z_1) \\ v(z_2) \\ \vdots \\ v(z_k) \end{bmatrix}, \quad [\mathcal{Z}]_v = \begin{bmatrix} [z_1]_v \\ [z_1, z_2]_v \\ \vdots \\ [z_1, \dots, z_k]_v \end{bmatrix}.$$

**Lemma 3.1.** Let  $\mathcal{Z} = \{z_1, \dots, z_k\}$  be an ordered set of distinct points, and let  $\Phi(\mathcal{Z})$  be defined as in (3.2). Then:

$$(3.4) \quad \Phi(\mathcal{Z})D(\mathcal{Z}) = J(\mathcal{Z})\Phi(\mathcal{Z}) \quad \text{and} \quad \Phi(\mathcal{Z})v(\mathcal{Z}) = [\mathcal{Z}]_v.$$

Moreover, if the function  $v(z)$  is analytic at  $z_0 \in \mathbb{C}$ , with the Taylor series  $v(z) = \sum_{k=0}^{\infty} v_k(z - z_0)^k$ , then

$$(3.5) \quad \lim_{z_1, \dots, z_k \rightarrow z_0} \Phi(\mathcal{Z})v(\mathcal{Z}) = \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_{k-1} \end{bmatrix}.$$

**Proof:** Formulas (3.4) are verified by direct computation; formula (3.5) follows from (3.4), since

$$\lim_{z_i \rightarrow z_0, i=1, \dots, j} [z_1, \dots, z_j]_v = \frac{v^{(j-1)}(z_0)}{(j-1)!} = v_{j-1}, \quad j = 1, \dots, k.$$

In the sequel it will be convenient to use the following notation:  $\text{diag}(X_1, \dots, X_k)$  stands for the block diagonal matrix with the diagonal blocks  $X_1, \dots, X_k$  (in that order).

**Lemma 3.2.** Let  $w_1, \dots, w_k$  be distinct points in  $\mathbb{D}$ , and let  $K$  be the unique solution of the Stein equation

$$(3.6) \quad K - AKA^* = EE^*,$$

where

$$(3.7) \quad A = \begin{bmatrix} J_{r_1}(w_1) & & & \\ & \ddots & & \\ & & J_{r_k}(w_k) & \end{bmatrix}, \quad E = \begin{bmatrix} E_{r_1} \\ \vdots \\ E_{r_k} \end{bmatrix}$$

and  $E_{r_j}$  are defined via the first formula in (3.1). Then the normalized Blaschke product

$$(3.8) \quad b(z) = e^{i\alpha} \prod_{j=1}^k \left( \frac{z - w_j}{1 - zw_j^*} \right)^{r_j}, \quad b(1) = 1,$$

admits a realization

$$(3.9) \quad b(z) = 1 + (z - 1)E^*(I - zA^*)^{-1}K^{-1}(I - A)^{-1}E,$$

and the following formula holds:

$$(3.10) \quad 1 - b(z)b(w)^* = (1 - zw^*)E^*(I - zA^*)^{-1}K^{-1}(I - w^*A)^{-1}E, \quad z, w \in \mathbb{D}.$$

**Proof:** First we note that  $K$  is given by the convergent series

$$(3.11) \quad K = \sum_{j=0}^{\infty} A^j E E^* (A^*)^j.$$

Since the pair  $(E^*, A^*)$  is observable, i.e.,  $\cap_{j=0}^{\infty} \text{Ker}(E^*(A^*)^j) = \{0\}$ , the matrix  $K$  is positive definite. Equality (3.10) follows from (3.9) and (3.6) by a standard straightforward calculation (or from (2.7) upon setting  $J = I_2$  in (2.5)–(2.7)). It follows from (3.10) that the function  $b(z)$  defined via (3.9) is inner. Using the fact that  $\det(I + XY) = \det(I + YX)$  for matrices  $X$  and  $Y$  of sizes  $u \times v$  and  $v \times u$  respectively, we obtain from (3.6) and (3.9):

$$\begin{aligned} b(z) &= \det(I + (z - 1)(I - zA^*)^{-1}K^{-1}(I - A)^{-1}(K - AKA^*)) \\ &= \det((I - zA^*)^{-1}K^{-1}(I - A)^{-1}) \\ &\quad \times \det((I - A)K(I - zA^*) + (z - 1)(K - AKA^*)) \\ &= \det((I - zA^*)^{-1}K^{-1}(I - A)^{-1}) \cdot \det((zI - A)K(I - A^*)) \\ &= c \frac{\det(zI - A)}{\det(I - zA^*)} \quad \text{for some } c \in \mathbb{C}, |c| = 1. \end{aligned}$$

It follows that the degree of  $b(z)$  is equal to  $r := \sum_{j=1}^k r_j$ , and  $b(z)$  has zeros at  $w_1, \dots, w_k$  of multiplicities  $r_1, \dots, r_k$ , respectively. Since  $b(1) = 1$ , the function  $b(z)$  indeed coincides with (3.8).  $\square$

The result of Lemma 3.2 is known also for matrix valued inner functions (see [5, Section 7.4], where it is given in a slightly different form for  $J$ -unitary matrix functions), in which case it can be interpreted as a formula for such functions having a prescribed left null pair with respect to the unit disk  $\mathbb{D}$  (see [5] and relevant references there).

Let

$$(3.12) \quad f(z) = \begin{cases} \frac{S(z)}{b(z)} & \text{if } z \notin \{z_1, \dots, z_\ell, w_{t+1}, \dots, w_k\} \\ f_j & \text{if } z = z_j, j = 1, \dots, \ell \end{cases}$$

where  $S(z)$  is a Schur function not vanishing at any of the  $z_j$ 's,  $b(z)$  is the Blaschke product given by (3.8). The points  $z_1, \dots, z_\ell$  are assumed to be distinct points in  $\mathbb{D}$ , and  $w_1, \dots, w_k$  are also assumed to be distinct in  $\mathbb{D}$ . Furthermore, we assume that

$$w_j = z_j \quad \text{for } j = 1, \dots, t; \quad \{w_{t+1}, \dots, w_k\} \cap \{z_{t+1}, \dots, z_\ell\} = \emptyset,$$

and

$$\frac{S(z_j)}{b(z_j)} \neq f_j \quad \text{for } j = t+1, \dots, \ell.$$

The cases when  $t = 0$ , i.e.,  $\{w_1, \dots, w_k\} \cap \{z_1, \dots, z_\ell\} = \emptyset$ , and when  $t = \min\{k, \ell\}$  are not excluded; in these cases the subsequent arguments should be modified in obvious ways. We let  $N = 2\ell + \sum_{j=1}^k r_j$ .

Take  $N$  distinct points in the unit disk sorted into  $k+2$  ordered sets  
(3.13)

$$\mathcal{M}_j = \{\mu_{j,1}, \dots, \mu_{j,r_j}\}, \quad j = 1, \dots, k; \quad \mathcal{N} = \{\nu_1, \dots, \nu_\ell\}; \quad \mathcal{Z} = \{z_1, \dots, z_\ell\}$$

and such that  $\mathcal{M}_j \cap \mathcal{W} = \emptyset$ ,  $j = 1, \dots, k$ , and  $\mathcal{N} \cap \mathcal{W} = \emptyset$ , where  $\mathcal{W} = \{w_1, \dots, w_k\}$ . Consider the corresponding Pick matrix

$$P = P_N(f; \mu_{1,1}, \dots, \mu_{k,r_k}, \nu_1, \dots, \nu_\ell, z_1, \dots, z_\ell).$$

We shall show that if  $\mu_{j,i}$  and  $\nu_j$  are sufficiently close to  $w_j$  and  $z_j$ , respectively, then  $\text{sq\_} P \geq N - \ell$ .

It is readily seen that  $P$  is a unique solution of the Stein equation

$$(3.14) \quad P - TPT^* = G_N G_N^* - CC^*$$

where  $G_N \in \mathbb{C}^N$  is defined via the second formula in (3.1) and

$$T = \begin{bmatrix} D(\mathcal{M}_1) & & & \\ & \ddots & & \\ & & D(\mathcal{M}_k) & \\ & & & D(\mathcal{N}) \\ & & & & D(\mathcal{Z}) \end{bmatrix}, \quad C = \begin{bmatrix} f(\mathcal{M}_1) \\ \vdots \\ f(\mathcal{M}_k) \\ f(\mathcal{N}) \\ f(\mathcal{Z}) \end{bmatrix}.$$

We recall that by definition (3.3) and in view of (3.12),

$$f(\mathcal{M}_j) = \begin{bmatrix} f(\mu_{j,1}) \\ \vdots \\ f(\mu_{j,r_j}) \end{bmatrix}, \quad f(\mathcal{N}) = \begin{bmatrix} f(\nu_1) \\ \vdots \\ f(\nu_\ell) \end{bmatrix}, \quad f(\mathcal{Z}) = \begin{bmatrix} f_1 \\ \vdots \\ f_\ell \end{bmatrix}.$$

Consider the matrices

$$B_j = \Phi(\mathcal{M}_j)\text{diag}(b(\mu_{j,1}), \dots, b(\mu_{j,r_j})) \quad (j = 1, \dots, k)$$

and note that by Lemma 3.1,

$$\begin{aligned} B_j D(\mathcal{M}_j) &= \Phi(\mathcal{M}_j) D(\mathcal{M}_j) \text{diag}(b(\mu_{j,1}), \dots, b(\mu_{j,r_j})) \\ &= J(\mathcal{M}_j) \Phi(\mathcal{M}_j) \text{diag}(b(\mu_{j,1}), \dots, b(\mu_{j,r_j})) \\ &= J(\mathcal{M}_j) B_j, \\ B_j f(\mathcal{M}_j) &= \Phi(\mathcal{M}_j) S(\mathcal{M}_j) = [\mathcal{M}_j]_S, \\ B_j G_{r_j} &= \Phi(\mathcal{M}_j) b(\mathcal{M}_j) = [\mathcal{M}_j]_b. \end{aligned}$$

The three last equalities together with block structure of the matrices  $T$ ,  $C$  and  $G_N$ , lead to

$$(3.15) \quad BT = T_1 B, \quad Y_1 = BY, \quad \text{and} \quad C_1 = BC,$$

where

$$(3.16)$$

$$B = \text{diag}(B_1, \dots, B_j, I_{2\ell}), \quad T_1 = \text{diag}(J(\mathcal{M}_1), \dots, J(\mathcal{M}_k), D(\mathcal{N}), D(\mathcal{Z})),$$

$$(3.17) \quad Y_1 = \begin{bmatrix} [\mathcal{M}_1]_b \\ \vdots \\ [\mathcal{M}_k]_b \\ G_{2\ell} \end{bmatrix} \quad \text{and} \quad C_1 = \begin{bmatrix} [\mathcal{M}_1]_S \\ \vdots \\ [\mathcal{M}_k]_S \\ f(\mathcal{N}) \\ f(\mathcal{Z}) \end{bmatrix}.$$

Pre- and post-multiplying (3.14) by  $B$  and  $B^*$ , respectively, we conclude, on account of (3.15), that the matrix  $P_1 := BPB^*$  is the unique solution of the Stein equation

$$(3.18) \quad P_1 - T_1 P_1 T_1^* = Y_1 Y_1^* - C_1 C_1^*,$$

where  $T_1$ ,  $Y_1$ , and  $C_1$  are given by (3.16) and (3.17).

Recall that all the entries in (3.18) depend on the  $\mu_{j,i}$ 's. We now let  $\mu_{j,i} \rightarrow w_j$ , for  $i = 1, \dots, r_j$ ,  $j = 1, \dots, k$ . Since  $b$  has zero of order  $r_j$  at  $w_j$ , it follows by (3.5) that

$$\lim_{\mu_{j,i} \rightarrow w_j} [\mathcal{M}_j]_b = 0, \quad j = 1, \dots, k,$$

and thus,

$$Y_2 := \lim_{\mu_{j,i} \rightarrow w_j} Y_1 = \begin{bmatrix} 0 \\ G_{2\ell} \end{bmatrix} \in \mathbb{C}^N.$$

Similarly, we get by (3.3),

$$C_2 := \lim_{\mu_{j,i} \rightarrow w_j} C_1 = \begin{bmatrix} \widehat{S}_1 \\ \vdots \\ \widehat{S}_k \\ f(\mathcal{N}) \\ f(\mathcal{Z}) \end{bmatrix}, \quad \text{where} \quad \widehat{S}_j = \begin{bmatrix} S(w_j) \\ \frac{S'(w_j)}{1!} \\ \vdots \\ \frac{S^{(r_j-1)}(w_j)}{(r_j-1)!} \end{bmatrix} \equiv \begin{bmatrix} s_{j,0} \\ s_{j,1} \\ \vdots \\ s_{j,r_j-1} \end{bmatrix}.$$

Furthermore, taking the limit in (3.16) as  $\mu_{j,i} \rightarrow w_j$ , we get

$$\begin{aligned} T_2 := \lim_{\mu_{j,i} \rightarrow w_j} T_1 &= \text{diag}(J_{r_1}(w_1), \dots, J_{r_k}(w_k), D(\mathcal{N}), D(\mathcal{Z})) \\ &= \text{diag}(A, D(\mathcal{N}), D(\mathcal{Z})), \end{aligned}$$

where  $A$  is given in (3.7). Since the above three limits exist, there also exists the limit

$$(3.19) \quad P_2 := \lim_{\mu_{j,i} \rightarrow w_j} P_1,$$

which serves to define a unique solution  $P_2$  of the Stein equation

$$(3.20) \quad P_2 - T_2 P_2 T_2^* = Y_2 Y_2^* - C_2 C_2^*.$$

Let  $\mathcal{S}_j$  be the lower triangular Toeplitz matrices defined by:

$$\mathcal{S}_j = \begin{bmatrix} s_{j,0} & 0 & \dots & 0 & 0 \\ s_{j,1} & s_{j,0} & \dots & 0 & 0 \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ s_{j,r_j-2} & & \dots & s_{j,0} & 0 \\ s_{j,r_j-1} & s_{j,r_j-2} & \dots & s_{j,1} & s_{j,0} \end{bmatrix}, \quad j = 1, \dots, k,$$

which are invertible because  $S(w_j) = s_{j,0} \neq 0$ ,  $j = 1, \dots, k$ . Let

$$R := \text{diag}(\mathcal{S}_1^{-1}, \dots, \mathcal{S}_k^{-1}, I_{2\ell}) \quad \text{and} \quad C_3 = \begin{bmatrix} E \\ f(\mathcal{N}) \\ f(\mathcal{Z}) \end{bmatrix},$$

where  $E$  is given in (3.7). The block structure of matrices  $R$ ,  $T_2$ ,  $C_2$ ,  $C_3$ ,  $E$  and  $Y_2$  together with selfevident relations

$$\mathcal{S}_j^{-1} J_{r_j}(w_j) = J_{r_j}(w_j) \mathcal{S}_j^{-1}, \quad \mathcal{S}_j^{-1} \widehat{S}_1 = E_{r_j} \quad (j = 1, \dots, k),$$

lead to

$$RT_2 = T_2R, \quad RC_2 = C_3 \quad \text{and} \quad RY_2 = Y_2.$$

Taking into account the three last equalities, we pre- and post-multiply (3.20) by  $R$  and  $R^*$ , respectively, to conclude that the matrix  $P_3 := RP_2R^*$  is the unique solution of the Stein equation

$$(3.21) \quad P_3 - T_2 P_3 T_2^* = Y_2 Y_2^* - C_3 C_3^*.$$

Denote

$$\begin{aligned} \mathcal{N}_1 &= \{\nu_1, \dots, \nu_t\}, & \mathcal{Z}_1 &= \{z_1, \dots, z_t\}, \\ \mathcal{N}_2 &= \{\nu_{t+1}, \dots, \nu_\ell\}, & \mathcal{Z}_2 &= \{z_{t+1}, \dots, z_\ell\}. \end{aligned}$$

By the hypothesis,  $\mathcal{Z}_2 \cap \mathcal{W} = \emptyset$ . Let  $\nu_j \rightarrow z_j$  for  $j = t+1, \dots, \ell$ . Then

$$f(\mathcal{N}_2) \rightarrow \frac{S}{b}(\mathcal{Z}_2),$$

and note that by the assumptions on  $f$ ,

$$(3.22) \quad \frac{S}{b}(z_j) \neq f(z_j), \quad j = t+1, \dots, \ell.$$

Taking limits in (3.21) as  $\nu_j \rightarrow z_j$  ( $j = t+1, \dots, \ell$ ), we arrive at the equality

$$(3.23) \quad P_4 - T_3 P_4 T_3^* = Y_2 Y_2^* - C_4 C_4^*, \quad P_4 := \lim_{\nu_j \rightarrow z_j, j=t+1, \dots, \ell} P_3,$$

where

$$T_3 = \text{diag}(A, D(\mathcal{N}_1), D(\mathcal{Z}_2), D(\mathcal{Z}_1), D(\mathcal{Z}_2)), \quad C_4 = \begin{bmatrix} E \\ f(\mathcal{N}_1) \\ \frac{S}{b}(\mathcal{Z}_2) \\ f(\mathcal{Z}_1) \\ f(\mathcal{Z}_2) \end{bmatrix}.$$

By (3.22), the matrix

$$L := \text{diag}\left(\left(\frac{S}{b} - f\right)(z_{t+1}), \dots, \left(\frac{S}{b} - f\right)(z_\ell)\right)$$

is invertible. Let

$$(3.24) \quad T_4 = \begin{bmatrix} A & 0 & 0 & 0 \\ 0 & D(\mathcal{Z}_2) & 0 & 0 \\ 0 & 0 & D(\mathcal{Z}_1) & 0 \\ 0 & 0 & 0 & D(\mathcal{N}_1) \end{bmatrix}, \quad C_5 = \begin{bmatrix} E \\ G_{\ell-t} \\ f(\mathcal{Z}_1) \\ f(\mathcal{N}_1) \end{bmatrix}, \quad Y_3 = \begin{bmatrix} 0 \\ G_{2t} \end{bmatrix}$$

and

$$X = \begin{bmatrix} I_{N-2\ell} & 0 & 0 & 0 & 0 \\ 0 & 0 & L^{-1} & 0 & -L^{-1} \\ 0 & 0 & 0 & I_t & 0 \\ 0 & I_t & 0 & 0 & 0 \end{bmatrix}.$$

Since

$$XT_3 = T_4X, \quad XC_4 = C_5, \quad XY_2 = Y_3,$$

pre- and post-multiplication of (3.23) by  $X$  and by  $X^*$ , respectively, leads to the conclusion that the matrix  $P_5 := X P_4 X^*$  is the unique solution of the Stein equation

$$(3.25) \quad P_5 - T_4 P_5 T_4^* = Y_3 Y_3^* - C_5 C_5^*.$$

By (3.24) and (3.25),  $P_5$  has the form

$$P_5 = \left[ \begin{array}{c|ccc} -K & Q_1^* & Q_2^* & Q_3^* \\ \hline Q_1 & R_{11} & R_{12} & R_{13} \\ Q_2 & R_{21} & R_{22} & R_{23} \\ Q_3 & R_{31} & R_{32} & R_{33} \end{array} \right],$$

where  $K$  is the matrix given by (3.11), and  $Q_j$ ,  $j = 1, 2, 3$ , are solutions of

$$\begin{aligned} Q_1 - D(\mathcal{Z}_2)Q_1 A^* &= -G_{\ell-t} E^*, \\ Q_2 - D(\mathcal{Z}_1)Q_2 A^* &= -f(\mathcal{Z}_1)E^*, \\ Q_3 - D(\mathcal{N}_1)Q_3 A^* &= -f(\mathcal{N}_1)E^*, \end{aligned}$$

respectively. Thus, denoting by  $[Q_\alpha]_j$  the  $j$ th row of  $Q_\alpha$ ,  $\alpha = 1, 2, 3$ , we have from the three last equalities (recall that  $w_j = z_j$  for  $j = 1, \dots, t$ ):

$$\begin{aligned} [Q_1]_j &= -E^*(I - z_{j+t} A^*)^{-1}, \quad j = 1, \dots, \ell - t; \\ [Q_2]_j &= -f_j E^*(I - w_j A^*)^{-1}, \quad j = 1, \dots, t; \\ [Q_3]_j &= -f(\nu_j) E^*(I - \nu_j A^*)^{-1}, \quad j = 1, \dots, t. \end{aligned}$$

Furthermore, in view of the three last relations and (3.10), and since  $b(w_j) = 0$ , following equalities are obtained:

$$\begin{aligned} [Q_1]_j K^{-1} [Q_1]_i^* &= \frac{1 - b(z_{t+j})b(z_{t+i})^*}{1 - z_{t+j}z_{t+i}^*}, \quad j, i = 1, \dots, \ell - t, \\ [Q_2]_j K^{-1} [Q_1]_i^* &= \frac{f_j}{1 - w_j z_{t+i}^*}, \quad j = 1, \dots, t; i = 1, \dots, \ell - t, \\ [Q_3]_j K^{-1} [Q_1]_i^* &= f(\nu_j) \frac{1 - b(\nu_j)b(z_{t+i})^*}{1 - \nu_j z_{t+i}^*}, \quad j = 1, \dots, t; i = 1, \dots, \ell - t, \\ [Q_2]_j K^{-1} [Q_2]_i^* &= \frac{f_j f_i^*}{1 - w_j w_i^*}, \quad j, i = 1, \dots, t, \\ [Q_3]_j K^{-1} [Q_2]_i^* &= \frac{f(\nu_j) f_i^*}{1 - \nu_j w_i^*}, \quad j, i = 1, \dots, t, \\ [Q_3]_j K^{-1} [Q_3]_i^* &= f(\nu_j) f(\nu_i)^* \frac{1 - b(\nu_j)b(\nu_i)^*}{1 - \nu_j \nu_i^*}, \quad j, i = 1, \dots, t. \end{aligned}$$

Next, the Schur complements are computed:

$$\begin{aligned} [R_{11} + Q_1 K^{-1} Q_1^*]_{ji} &= \frac{-1}{1 - z_{t+j} z_{t+i}^*} + \frac{1 - b(z_{t+j}) b(z_{t+i})^*}{1 - z_{t+j} z_{t+i}^*} \\ &= -\frac{b(z_{t+j}) b(z_{t+i})^*}{1 - z_{t+j} z_{t+i}^*}, \end{aligned}$$

$$[R_{21} + Q_2 K^{-1} Q_1^*]_{ji} = 0,$$

$$\begin{aligned} [R_{31} + Q_3 K^{-1} Q_1^*]_{ji} &= -\frac{f(\nu_j)}{1 - \nu_j z_{t+i}^*} + f(\nu_j) \frac{1 - b(\nu_j) b(z_{t+i})^*}{1 - \nu_j z_{t+i}^*} \\ &= -f(\nu_j) \frac{b(\nu_j) b(z_{t+i})^*}{1 - \nu_j z_{t+i}^*} \\ &= -\frac{S(\nu_j) b(z_{t+i})^*}{1 - \nu_j z_{t+i}^*}, \end{aligned}$$

$$[R_{22} + Q_2 K^{-1} Q_2^*]_{ji} = \frac{1 - f_j f_i^*}{1 - w_j w_i^*} + \frac{f_j f_i^*}{1 - w_j w_i^*} = \frac{1}{1 - w_j w_i^*},$$

$$[R_{32} + Q_3 K^{-1} Q_2^*]_{ji} = \frac{1 - f(\nu_j) f_i^*}{1 - \nu_j w_i^*} + \frac{f(\nu_j) f_i^*}{1 - \nu_j w_i^*} = \frac{1}{1 - \nu_j w_i^*},$$

$$\begin{aligned} [R_{33} + Q_3 K^{-1} Q_3^*]_{ji} &= \frac{1 - f(\nu_j) f(\nu_i)^*}{1 - \nu_j \nu_i^*} + f(\nu_j) f(\nu_i)^* \frac{1 - b(\nu_j) b(\nu_i)^*}{1 - \nu_j \nu_i^*} \\ &= \frac{1 - S(\nu_j) S(\nu_i)^*}{1 - \nu_j \nu_i^*}. \end{aligned}$$

In the obtained Schur complement  $[R_{i,j} + Q_i K^{-1} Q_j]_{i,j=1}^3$ , we let  $\nu_j \rightarrow w_j$  ( $j = 1, 2, \dots, t$ ) and then pre- and post-multiply by the matrix  $\begin{bmatrix} b(\mathcal{Z}_2)^{-1} & 0 \\ 0 & I_{2t} \end{bmatrix}$  and by its adjoint, respectively, resulting in the matrix

$$(3.26) \quad \begin{bmatrix} -K_{11} & 0 & K_{31}^* \\ 0 & K_{22} & K_{22} \\ K_{31} & K_{22} & \left[ \frac{1 - S(w_j) S(w_i)^*}{1 - w_j w_i^*} \right]_{j,i=1}^t \end{bmatrix},$$

where

$$(3.27) \quad K_{11} = \left[ \frac{1}{1 - z_{t+j} z_{t+i}^*} \right]_{j,i=1}^{\ell-t}, \quad K_{22} = \left[ \frac{1}{1 - w_j w_i^*} \right]_{j,i=1}^t, \quad K_{31} = \left[ -\frac{S(w_j)}{1 - w_j z_{t+i}^*} \right]_{j,i=1}^{t,\ell-t}.$$

Note that the  $j$ -th row  $[K_{31}]_j$  of  $K_{31}$  can be written in the form

$$[K_{31}]_j = S(w_j) G_{\ell-t}^*(I - w_j D(\mathcal{Z}_2)^*)^{-1} \quad (j = 1, \dots, t).$$

Using Lemma 2.2, in view of the reductions made so far from  $P$  to  $P_5$ , to prove that  $\text{sq-}P \geq N - \ell$ , we only have to show that the Schur complement  $\mathbf{S}$  to the block  $\begin{bmatrix} -K_{11} & 0 \\ 0 & K_{22} \end{bmatrix}$  in (3.26) has at least  $t$  negative eigenvalues, i.e., is negative

definite. This Schur complement is equal to

$$\begin{aligned}
 \mathbf{S} &= \left[ \frac{1 - S(w_j)S(w_i)^*}{1 - w_j w_i^*} \right]_{j,i=1}^t + K_{31} K_{11}^{-1} K_{31}^* - K_{22} \\
 &= - \left[ \frac{S(w_j)S(w_i)^*}{1 - w_j w_i^*} \right]_{j,i=1}^t + K_{31} K_{11}^{-1} K_{31}^* \\
 &= - \left[ \frac{S(w_j)S(w_i)^*}{1 - w_j w_i^*} - [K_{31}]_j K_{11}^{-1} ([K_{31}]_i)^* \right]_{j,i=1}^t \\
 &= - \left[ \frac{S(w_j)S(w_i)^*}{1 - w_j w_i^*} \right. \\
 (3.28) \quad &\left. - S(w_j)G_{\ell-t}(I - w_j D(\mathcal{Z}_2)^*)^{-1} K_{11}^{-1} (I - w_i^* D(\mathcal{Z}_2))^{-1} G_{\ell-t}^* S(w_i)^* \right]_{j,i=1}^t.
 \end{aligned}$$

Introduce the rational function

$$(3.29) \quad \vartheta(z) = 1 + (z - 1)G_{\ell-t}^*(I - zD(\mathcal{Z}_2)^*)^{-1}K_{11}^{-1}(I - D(\mathcal{Z}_2))^{-1}G_{\ell-t}.$$

and note that  $K_{11}$  satisfies the Stein equation

$$K_{11} - D(\mathcal{Z}_2)K_{11}D(\mathcal{Z}_2)^* = G_{\ell-t}G_{\ell-t}^*.$$

Upon setting  $k = \ell - t$  and  $r_1 = \dots = r_k = 1$  and picking points  $z_{t+1}, \dots, z_\ell$  instead of  $w_1, \dots, w_k$  in Lemma 3.2, we get  $A = D(\mathcal{Z}_2)$ ,  $K = K_{11}$ ,  $E = G_{\ell-t}$  and thus, by Lemma 3.2, we conclude that

$$(3.30) \quad \vartheta(z) = e^{i\alpha} \prod_{j=1}^{\ell-t} \frac{z - z_{t+j}}{1 - zz_{t+j}^*},$$

where  $\alpha \in \mathbb{R}$  is chosen to provide the normalization  $\vartheta(1) = 1$ , and moreover, relation (3.10) in the present setting takes the form

$$(3.31) \quad 1 - \vartheta(z)\vartheta(w)^* = (1 - zw^*)G_{\ell-t}^*(I - zD(\mathcal{Z}_2)^*)^{-1}K_{11}^{-1}(I - w^*D(\mathcal{Z}_2))^{-1}G_{\ell-t}.$$

It follows from (3.30) that  $Z(\vartheta) = \mathcal{Z}_2$  (more precisely,  $\vartheta$  is of degree  $\ell - t$  and has simple poles at  $z_{t+1}, \dots, z_\ell$ ). Since  $\mathcal{Z}_1 \cap \mathcal{Z}_2 = \emptyset$ , it follows that  $\vartheta(w_j) \neq 0$ ,  $j = 1, \dots, t$ . Upon making use of (3.31), we rewrite (3.28) as

$$(3.32) \quad \mathbf{S} = - \left[ \frac{S(w_j)\vartheta(w_j)\vartheta(w_i)^*S(w_i)^*}{1 - w_j w_i^*} \right]_{j,i=1}^t$$

and conclude, since  $S(w_j)\vartheta(w_j) \neq 0$  ( $j = 1, \dots, t$ ) that  $\mathbf{S}$  is negative definite. Summarizing, we have

$$\begin{aligned}
 \text{sq}_- P \geq \text{sq}_- P_5 &= \text{sq}_-(-K) + \text{sq}_- \begin{bmatrix} -K_{11} & 0 \\ 0 & K_{22} \end{bmatrix} + \text{sq}_- \mathbf{S} \\
 &= (N - 2\ell) + (\ell - t) + t = N - \ell,
 \end{aligned}$$

which completes the proof.  $\square$

#### 4. Theorems 1.7 and 1.11: proofs

**Proof of Theorem 1.7:** The implication  $3 \Rightarrow 1$  was already proved, and  $1 \Rightarrow 3$  follows from the definition of  $\mathcal{S}_\kappa$ .

Assume 2 holds, and let  $\tilde{f}$  be the standard function that extends  $f$  as stated in 2. By Theorem 1.8  $\tilde{f} \in \mathcal{S}_\kappa$ . It is obvious from the definition of  $\mathcal{S}_\kappa$  that we have

$f \in \mathcal{S}_{\kappa'}$  for some  $\kappa' \leq \kappa$ . However, Remark 1.10 implies that in fact  $\kappa' = \kappa$ , and **1** follows.

It remains to prove **3  $\Rightarrow$  2**. Assume that  $f$  satisfies **3**. Arguing as in the proof of **3.  $\Rightarrow$  1**, we obtain the meromorphic function  $F(z)$  given by (2.16) such that  $F(z) = f(z)$  for  $z \in \mathbb{D} \setminus (\mathcal{Z} \cup \Lambda)$  (in the notation of the proof of **3  $\Rightarrow$  1**). By the already proved part of Theorem 1.7, we know that  $f \in \mathcal{S}_\kappa$ . By the second statement of Lemma 2.3,  $\tilde{F} \in \mathcal{S}_{\kappa'}$  for some  $\kappa' \leq \kappa + n$ , where  $\tilde{F}$  is the restriction of  $F$  to the set  $\mathbb{D} \setminus (\mathcal{Z} \cup \Lambda)$ . By continuity (statement 3. of Lemma 2.2), the function  $F$ , considered on its natural domain of definition  $\mathbb{D} \setminus P(F)$ , where  $P(F)$  is the set of poles of  $F$ , also belongs to  $\mathcal{S}_{\kappa'}$ . Using Theorem 1.4, write  $F = \frac{S}{B}$ , where  $S$  is a Schur function and  $B$  is a Blaschke product of degree  $\kappa'$  without common zeros. Thus,  $f$  admits an extension to a standard function  $\tilde{f}$  with  $\kappa'$  poles and  $\rho$  jumps, for some nonnegative integer  $\rho$ . By Theorem 1.8 and Remark 1.10 we have

$$(4.1) \quad \mathbf{k}_m(f) = \kappa' + \rho, \quad \text{for every } m \geq \kappa' + 2\rho.$$

On the other hand, since  $f \in \mathcal{S}_\kappa$ , we have  $\mathbf{k}_n(f) = \kappa$  for all sufficiently large  $n$ . Comparing with (4.1), we see that  $\kappa' + \rho = \kappa$ , and **2** follows.

**Proof of Theorem 1.11:** If  $f \in \mathcal{S}_\kappa$ , and  $\tilde{f}$  is the standard function that extends  $f$  (as in statement **2** of Theorem 1.7), then by Theorem 1.8

$$N(f) \leq N(\tilde{f}) \leq 2\kappa.$$

If  $\mathbf{k}_{2\kappa}(f) = \mathbf{k}_{2\kappa+3}(f) = \kappa$ , then obviously **3** of Theorem 1.7 holds, and  $f \in \mathcal{S}_\kappa$  by Theorem 1.7.

## 5. A local result

Let  $\Omega \subseteq \mathbb{D}$  be an open set, and let

$$\mathbf{k}_n(f; \Omega) := \max_{z_1, \dots, z_n \in \Omega} \text{sq}_- P_n(f; z_1, \dots, z_n),$$

where the domain of definition of the function  $f$  contains  $\Omega$ . A known local result (see [4, Theorem 1.1.4], where it is proved in the more general setting of operator valued functions) states that for a meromorphic function  $f$  from the class  $\mathcal{S}_\kappa$  and for every open set  $\Omega \subseteq \mathbb{D}$  that does not contain any poles of  $f$ , there is an integer  $n$  such that  $\mathbf{k}_n(f; \Omega) = \kappa$ . However, the minimal such  $n$  can be arbitrarily large, in contrast with Theorem 1.8, as the following example shows.

**Example 5.1.** Fix a positive integer  $n$ , and let

$$f_n(z) = \frac{z^n(2-z)}{2z-1} = \frac{S(z)}{b(z)},$$

where  $S(z) = z^n$  is a Schur function, and  $b(z) = \frac{z-1/2}{1-z/2}$  is a Blaschke factor. By Theorem 1.4,  $f_n \in \mathcal{S}_1$ . Thus, the kernel

$$K(z, w) := \frac{1 - f_n(z)f_n(w)^*}{1 - zw^*} = 1 + zw^* + \dots + z^{n-1}(w^*)^{n-1} - \frac{3z^n(w^*)^n}{(2z-1)(2w^*-1)}$$

has one negative square on  $\mathbb{D} \setminus \{\frac{1}{2}\}$ . Select  $n$  distinct points  $\mathcal{Z} = \{z_1, \dots, z_n\}$  (in some order) near zero, and multiply the Pick matrix  $P_n(f; z_1, \dots, z_n)$  by  $\Phi(\mathcal{Z})$  on

the left and by  $\Phi(\mathcal{Z})^*$  on the right, where  $\Phi(\mathcal{Z})$  is defined in (3.2). By Lemma 3.1, (5.1)

$$\lim_{z_1, \dots, z_n \rightarrow 0} \Phi(\mathcal{Z}) P_n(f; z_1, \dots, z_n) \Phi(\mathcal{Z})^* = \left[ \frac{1}{i!} \frac{1}{j!} \left( \frac{\partial^{i+j}}{\partial z^i \partial (w^*)^j} K(z, w) \right)_{z, w=0} \right]_{i, j=0}^{n-1},$$

which is the identity matrix. Therefore, by Lemma 2.2 there exists a  $\delta_n > 0$  such that  $P_n(f; z_1, \dots, z_n) \geq 0$  if  $|z_1|, \dots, |z_n| < \delta_n$ .

On the other hand, since

$$\left( \frac{\partial^{2n}}{\partial z^n \partial (w^*)^n} K(z, w) \right)_{z, w=0} = -3,$$

selecting an ordered set of  $n+1$  distinct points  $\mathcal{Z}' = \{z_1, \dots, z_{n+1}\}$  in a neighborhood of zero, analogously to (5.1) we obtain

$$\begin{aligned} & \lim_{z_1, \dots, z_{n+1} \rightarrow 0} \Phi(\mathcal{Z}') P_{n+1}(f; z_1, \dots, z_{n+1}) \Phi(\mathcal{Z}')^* \\ &= \left[ \frac{1}{i!} \frac{1}{j!} \left( \frac{\partial^{i+j}}{\partial z^i \partial (w^*)^j} K(z, w) \right)_{z, w=0} \right]_{i, j=0}^n = \begin{bmatrix} I_n & 0 \\ 0 & -3 \end{bmatrix}. \end{aligned}$$

By Lemma 2.2 again, there exists a  $\delta'_n > 0$  such that  $P_{n+1}(f; z_1, \dots, z_{n+1})$  has exactly one negative eigenvalue if  $|z_1|, \dots, |z_{n+1}| < \delta'_n$ . (Note that  $P_{n+1}(f; z_1, \dots, z_{n+1})$  cannot have more than one negative eigenvalue because  $f \in \mathcal{S}_1$ .)

Theorem 1.7, or more precisely its proof, allows us to extend the local result to the more general classes  $\mathcal{S}_\kappa$ :

**Theorem 5.2.** *If  $f \in \mathcal{S}_\kappa$  is defined on  $\mathbb{D} \setminus \Lambda$ , where  $\Lambda$  is a discrete set, then for every open set  $\Omega \subseteq \mathbb{D} \setminus \Lambda$  there exists a positive integer  $n$  such that*

$$(5.2) \quad \mathbf{k}_n(f; \Omega) = q + \ell_\Omega,$$

where  $\ell_\Omega$  is the number of jump points of  $f$  that belong to  $\Omega$ , and  $q$  is the number of poles of  $f$  in  $\mathbb{D}$ , counted with multiplicities.

Note that in view of **1  $\Leftrightarrow$  2** of Theorem 1.7, the right hand side of (5.2) is equal to  $\kappa - \ell_{\mathbb{C}\Omega}$ , where  $\ell_{\mathbb{C}\Omega}$  is the number of jump points of  $f$  that do not belong to  $\Omega$ . Therefore, since  $\mathbf{k}_n(f; \Omega)$  is completely independent of the values of  $f$  at jump points outside  $\Omega$ , it is easy to see that

$$(5.3) \quad \mathbf{k}_m(f; \Omega) \leq q + \ell_\Omega \quad \text{for every integer } m > 0.$$

**Proof.** We may assume by Theorem 1.7 that  $f$  is a standard function. We may also assume that  $f$  has no jumps outside  $\Omega$ , because the values of  $f$  at jump points outside  $\Omega$  do not contribute to  $\mathbf{k}_n(f; \Omega)$ .

Let  $w_1, \dots, w_k$  be the distinct poles of  $f$  in  $\Omega$  (if  $f$  has no poles in  $\Omega$ , the subsequent argument is simplified accordingly), of orders  $r_1, \dots, r_k$ , respectively, and let  $z_1, \dots, z_\ell$  be the jumps of  $f$ . Then analogously to (3.12) we have

$$(5.4) \quad f(z) = \begin{cases} \frac{\widehat{S}(z)}{b(z)} & \text{if } z \notin \{z_1, \dots, z_\ell, w_{t+1}, \dots, w_k\} \\ f_j & \text{if } z = z_j, j = 1, \dots, \ell \end{cases}$$

where  $b(z)$  is the Blaschke product having zeros  $w_1, \dots, w_k$  of orders  $r_1, \dots, r_k$ , respectively, given by (3.8). We assume that

$$w_j = z_j \quad \text{for } j = 1, \dots, t; \quad \{w_{t+1}, \dots, w_k\} \cap \{z_{t+1}, \dots, z_\ell\} = \emptyset,$$

and

$$\frac{\widehat{S}(z_j)}{b(z_j)} \neq f_j \quad \text{for } j = t+1, \dots, \ell.$$

The function  $\widehat{S}(z)$  is of the form  $\widehat{S}(z) = S(z)/b_{\text{out}}(z)$ , where  $S(z)$  is a Schur function that does not vanish at  $w_1, \dots, w_k$ , and  $b_{\text{out}}(z)$  is the Blaschke product whose zeros coincide with the poles of  $f$  outside  $\Omega$ , with matched orders. Let  $N = 2\ell + \sum_{j=1}^k r_j$ , and select  $N$  distinct points as in (3.13), with the additional proviso that all these points belong to  $\Omega$ . Select also  $n$  distinct points  $\Xi = \{\xi_1, \dots, \xi_n\}$ ,  $\xi_j \in \Omega$ , disjoint from (3.13), in a vicinity of some  $z_0 \in \Omega$ ; we assume that  $f$  is analytic at  $z_0$ . The number  $n$  of points  $\xi_j$ , and further specifications concerning the set  $\Xi$ , will be determined later.

Let

$$P = P_{N+n}(f; \mu_{j,i}, \nu_1, \dots, \nu_\ell, z_1, \dots, z_\ell, \xi_1, \dots, \xi_n).$$

We now repeat the steps between (3.13) and (3.26) of the proof of Theorem 1.8, applied to the top left  $N \times N$  corner of  $P$ . As a result, we obtain the following matrix:

$$(5.5) \quad P_6 = \begin{bmatrix} -K_{11} & 0 & K_{31}^* & K_{41}^* \\ 0 & K_{22} & K_{22} & K_{42}^* \\ K_{31} & K_{22} & K_{33} & K_{43}^* \\ K_{41} & K_{42} & K_{43} & K_{44} \end{bmatrix},$$

where  $K_{11}$ ,  $K_{22}$ , and  $K_{31}$  are as in (3.27), and

$$\begin{aligned} K_{41} &= \left[ -\frac{\widehat{S}(\xi_j)}{1 - \xi_j z_{t+i}^*} \right]_{j,i=1}^{n,\ell-t}, & K_{42} &= \left[ \frac{1}{1 - \xi_j w_i^*} \right]_{j,i=1}^{n,\ell-t}, \\ K_{43} &= \left[ \frac{1 - \widehat{S}(\xi_j)\widehat{S}(w_i)^*}{1 - \xi_j w_i^*} \right]_{j,i=1}^{n,\ell-t}, & K_{44} &= \left[ \frac{1 - \widehat{S}(\xi_j)\widehat{S}(\xi_i)^*}{1 - \xi_j \xi_i^*} \right]_{j,i=1}^n, \\ K_{33} &= \left[ \frac{1 - \widehat{S}(w_j)\widehat{S}(w_i)^*}{1 - w_j w_i^*} \right]_{j,i=1}^t. \end{aligned}$$

The sizes of matrices  $K_{11}$ ,  $K_{22}$ ,  $K_{33}$ , and  $K_{44}$  are  $(\ell-t) \times (\ell-t)$ ,  $t \times t$ ,  $t \times t$ , and  $n \times n$ , respectively. It follows that

$$(5.6) \quad \text{sq\_}P \geq N - 2\ell + \text{sq\_}P_6 = \sum_{j=1}^k r_j + \text{sq\_}P_6.$$

Take the Schur complement  $\mathbf{S}$  to the block  $\begin{bmatrix} -K_{11} & 0 \\ 0 & K_{22} \end{bmatrix}$  in (5.5):

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} \\ \mathbf{S}_{21} & \mathbf{S}_{22} \end{bmatrix},$$

where

$$\mathbf{S}_{22} = K_{44} - K_{42}K_{22}^{-1}K_{42}^* + K_{41}K_{11}^{-1}K_{41}^*.$$

We have

$$(5.7) \quad \text{sq}_- P_6 = \ell - t + \text{sq}_- \mathbf{S}.$$

Analogously to formulas (3.28) and (3.32) we obtain

$$(5.8) \quad \begin{aligned} \mathbf{S}_{11} &= - \left[ \frac{\widehat{S}(w_j)\vartheta(w_j)\vartheta(w_i)^*\widehat{S}(w_i)^*}{1-w_jw_i^*} \right]_{j,i=1}^t, \\ \mathbf{S}_{21} &= - \left[ \frac{\widehat{S}(\xi_j)\vartheta(\xi_j)\vartheta(w_i)^*\widehat{S}(w_i)^*}{1-\xi_jw_i^*} \right]_{j,i=1}^{t,n}, \end{aligned}$$

where the rational function  $\vartheta$  is given by (3.29). Note that  $\widehat{S}(w_i)\vartheta(w_i) \neq 0$ , for  $i = 1, \dots, t$ . Multiply  $\mathbf{S}$  by the matrix

$$(5.9) \quad \text{diag} \left( \frac{1}{\widehat{S}(w_1)\vartheta(w_1)}, \dots, \frac{1}{\widehat{S}(w_t)\vartheta(w_t)}, I \right)$$

on the left and by the adjoint of (5.9) on the right. We arrive at

$$(5.10) \quad P_7 = \begin{bmatrix} - \left[ \frac{1}{1-w_jw_i^*} \right]_{j,i=1}^t & \left( - \left[ \frac{\widehat{S}(\xi_j)\vartheta(\xi_j)}{1-\xi_jw_i^*} \right]_{j,i=1}^{n,t} \right)^* \\ - \left[ \frac{\widehat{S}(\xi_j)\vartheta(\xi_j)}{1-\xi_jw_i^*} \right]_{j,i=1}^{n,t} & K_{44} - K_{42}K_{22}^{-1}K_{42}^* + K_{41}K_{11}^{-1}K_{41}^* \end{bmatrix}.$$

Finally, we take the Schur complement, denoted  $P_8$ , to the block  $- \left[ \frac{1}{1-w_jw_i^*} \right]_{j,i=1}^t$  of  $P_7$ . Then

$$(5.11) \quad \text{sq}_- \mathbf{S} = t + \text{sq}_- P_8.$$

The  $(j, i)$  entry of  $P_8$  is

$$\begin{aligned} &\frac{1 - \widehat{S}(\xi_j)\widehat{S}(\xi_i)^*}{1 - \xi_j\xi_i^*} - G_{\ell-t}^*(I - \xi_j D(\mathcal{W}_1)^*)^{-1} K_{22}^{-1} (I - \xi_i^* D(\mathcal{W}_1))^{-1} G_{\ell-t} \\ &+ \widehat{S}(\xi_j)G_{\ell-t}^*(I - \xi_j D(\mathcal{Z}_2)^*)^{-1} K_{11}^{-1} (I - \xi_i^* D(\mathcal{Z}_2))^{-1} G_{\ell-t} \widehat{S}(\xi_i)^* \\ &+ \widehat{S}(\xi_j)\vartheta(\xi_j)G_{\ell-t}^*(I - \xi_j D(\mathcal{W}_1)^*)^{-1} K_{22}^{-1} (I - \xi_i^* D(\mathcal{W}_1))^{-1} G_{\ell-t} \vartheta(\xi_i)^* \widehat{S}(\xi_i)^*. \end{aligned}$$

Since

$$K_{22} - D(\mathcal{W}_1)K_{22}D(\mathcal{W}_1)^* = G_{\ell-t}G_{\ell-t}^*,$$

we conclude (as it was done for the function  $\vartheta$  given in (3.29)) that the rational function

$$\widehat{\vartheta}(z) = 1 + (z - 1)G_{\ell-t}^*(I - zD(\mathcal{W}_1)^*)^{-1} K_{22}^{-1} (I - D(\mathcal{W}_1))^{-1} G_{\ell-t}$$

is inner, and its set of zeros is  $Z(\widehat{\vartheta}) = \mathcal{W}_1$ . Using (3.31) and the formula

$$1 - \widehat{\vartheta}(z)\widehat{\vartheta}(w)^* = G_{\ell-t}^*(I - zD(\mathcal{W}_1)^*)^{-1} K_{22}^{-1} (I - w^* D(\mathcal{W}_1))^{-1} G_{\ell-t}$$

similar to (3.31), the expression for the  $(j, i)$  entry of  $P_8$  takes the form

$$\begin{aligned} & \frac{1 - \widehat{S}(\xi_j) \widehat{S}(\xi_i)^*}{1 - \xi_j \xi_i^*} - \frac{1 - \widehat{\vartheta}(\xi_j) \widehat{\vartheta}(\xi_i)^*}{1 - \xi_j \xi_i^*} + \widehat{S}(\xi_j) \frac{1 - \vartheta(\xi_j) \vartheta(\xi_i)^*}{1 - \xi_j \xi_i^*} \widehat{S}(\xi_i)^* \\ & + \widehat{S}(\xi_j) \vartheta(\xi_j) \frac{1 - \widehat{\vartheta}(\xi_j) \widehat{\vartheta}(\xi_i)^*}{1 - \xi_j \xi_i^*} \vartheta(\xi_i)^* \widehat{S}(\xi_i)^* \\ & = \widehat{\vartheta}(\xi_j) \frac{1 - \widehat{S}(\xi_j) \vartheta(\xi_j) \vartheta(\xi_i)^* \widehat{S}(\xi_i)^*}{1 - \xi_j \xi_i^*} \widehat{\vartheta}(\xi_i)^*. \end{aligned}$$

We now assume that the point  $z_0 \in \Omega$  in a neighborhood of which the set  $\Xi$  is selected is such that  $\widehat{\vartheta}(z_0) \neq 0$ . Then we may assume that  $\widehat{\vartheta}(\xi_j) \neq 0$ ,  $j = 1, \dots, n$ . Multiply  $P_8$  on the left by the matrix  $\text{diag}(\widehat{\vartheta}(\xi_1)^{-1}, \dots, \widehat{\vartheta}(\xi_n)^{-1})$  and on the right by its adjoint, resulting in a matrix

$$P_9 = \left[ \frac{1 - T(\xi_j) T(\xi_i)^*}{1 - \xi_j \xi_i^*} \right]_{j,i=1}^n,$$

where the function  $T(z)$  is given by  $T(z) = \widehat{S}(z) \vartheta(z)$ . We have

$$(5.12) \quad \text{sq\_}_P_8 = \text{sq\_}_P_9$$

Note that  $T(z)$  is a meromorphic function with no poles in  $\Omega$ . By [4, Theorem 1.1.4], there exist a positive integer  $n$  and points  $\xi_1, \dots, \xi_n$  in a neighborhood of  $z_0$  such that  $P_9$  has  $q - \sum_{j=1}^k r_j$  (the number of poles of  $T(z)$ ) negative eigenvalues. Using this choice of  $\xi_j$ , and combining (5.6), (5.7), (5.11), and (5.12), we obtain

$$\text{sq\_}_P \geq q + \ell.$$

Now equality (5.2) follows in view of (5.3).  $\square$

## 6. An open problem

Fix an integer  $k \geq 3$ . An open set  $D \subseteq \mathbb{D}$  is said to have the  $k$ -th Hindmarsh property if every function  $f$  defined on  $D$  and such that the matrices  $P_k(f; z_1, \dots, z_k)$  are positive semidefinite for every  $k$ -tuple of points  $z_1, \dots, z_k \in D$ , admits a (necessarily unique) extension to a Schur function (defined on  $\mathbb{D}$ ). Theorem 1.1 shows that  $\mathbb{D}$  has the 3-rd Hindmarsh property. Example 5.1 shows that the open disk of radius  $\delta_n$  centered at the origin does not have the  $n$ -th Hindmarsh property, if  $\delta_n$  is sufficiently small.

Denote by  $\mathcal{H}_k$  the collection of all sets with the  $k$ -th Hindmarsh property. Clearly,

$$\mathcal{H}_3 \subseteq \mathcal{H}_4 \subseteq \dots \subseteq \mathcal{H}_k \subseteq \dots$$

**Proposition 6.1.** *Let  $D \in \mathcal{H}_k$ . If  $\mathcal{Z} \subset D$  is a discrete set in  $D$ , then  $D \setminus \mathcal{Z}$  also has the  $k$ -th Hindmarsh property.*

**Proof.** Let  $f$  be a function with the domain of definition  $D \setminus \mathcal{Z}$  and such that  $P_k(f; z_1, \dots, z_k) \geq 0$  for every set of distinct points  $z_1, \dots, z_k \in D \setminus \mathcal{Z}$ . In particular,  $P_3(f; z_1, z_2, z_3) \geq 0$  for every triple of distinct points  $z_1, z_2, z_3 \in D \setminus \mathcal{Z}$ . By Hindmarsh's theorem,  $f$  is analytic on  $D \setminus \mathcal{Z}$ . Also,

$$|f(z)|^2 = (|f(z)|^2 - 1) + 1 = -P_1(f; z)(1 - |z|^2) + 1 \leq 1$$

for every  $z \in D \setminus \mathcal{Z}$ . Thus,  $f$  admits an analytic continuation to a function  $\hat{f}$  on  $D$ . By continuity,  $P_k(\hat{f}; z_1, \dots, z_k) \geq 0$  for every  $k$ -tuple of distinct points  $z_1, \dots, z_k \in D$ . Since  $D \in \mathcal{H}_k$ ,  $\hat{f}$  admits an extension to a Schur function.  $\square$

**Corollary 6.2.** *Let  $D_0 = \mathbb{D}$ ,  $D_j = D_{j-1} \setminus \mathcal{Z}_{j-1}$ ,  $j = 1, 2, \dots$ , where  $\mathcal{Z}_{j-1}$  is a discrete set in  $D_{j-1}$ . Then all the sets  $D_j$ ,  $j = 1, 2, \dots$ , have the 3-rd Hindmarsh property.*

Using the sets with the Hindmarsh property, the implication **3.  $\Rightarrow$  1.** of Theorem 1.7 can be extended to a larger class of functions, as follows:

**Theorem 6.3.** *Let  $f$  be defined on  $D$ , where  $D \in \mathcal{H}_p$ . Then  $f$  belongs to  $\mathcal{S}_\kappa(D)$  if and only if*

$$(6.1) \quad \mathbf{k}_n(f) = \mathbf{k}_{n+p}(f) = \kappa$$

for some integer  $n$ .

The definition of  $\mathcal{S}_\kappa(D)$  is the same as  $\mathcal{S}_\kappa$ , with the only difference being that  $\mathcal{S}_\kappa(D)$  consists of functions defined on  $D$ . The proof of Theorem 6.3 is essentially the same as that of **3.  $\Rightarrow$  1.** of Theorem 1.7.

**Problem 6.4.** *Describe the structure of sets with the  $k$ -th Hindmarsh property.*

In the general case this seems to be a very difficult unsolved problem. We do not address this problem in the present paper.

## References

1. V. M. Adamyan, D. Z. Arov, and M. G. Krein. *Analytic properties of the Schmidt pairs of a Hankel operator and the generalized Schur-Takagi problem*, (Russian) Mat. Sb. (N.S.) **86(128)** (1971), 34–75.
2. J. Agler and N. J. Young. *Functions that are almost multipliers of Hilbert function spaces*, Proc. of London Math. Soc., **76** (1998), 453–475.
3. N. I. Akhiezer. *On a minimum problem in function theory and the number of roots of an algebraic equation inside the unit disc*, Izv. Akad. Nauk SSSR Mat. Estestv. Nauk **9** (1930), 1169–1189, English transl. in: *Topics in interpolation theory*, Oper. Theory Adv. Appl., **95**, 19–35.
4. D. Alpay, A. Dijksma, J. Rovnyak, and H. de Snoo. *Schur functions, operator colligations and reproducing kernel Pontryagin spaces*, OT96, Birkhäuser, 1997.
5. J. A. Ball, I. Gohberg, and L. Rodman. *Interpolation of rational matrix functions*. OT45. Birkhäuser Verlag, 1990.
6. J. A. Ball and J. W. Helton. *Interpolation problems of Pick–Hervanlinna and Loewner type for meromorphic matrix functions: parametrizations of the set of all solutions*, Integral Equations and Operator Theory **9** (1986), 155–203.
7. R. Bhatia. *Matrix analysis*, Springer–Verlag, New York, 1996.
8. V. Bolotnikov and L. Rodman. *Krein–Langer factorizations via pole triples*, Integral Equations and Operator Theory, to appear.
9. A. Dijksma, H. Langer, and H. S. de Snoo. *Characteristic functions of unitary operator colligations in  $\Pi_\kappa$  spaces*, Operator Theory: Advances and Applications, **19** (1986), 125–194.
10. W. F. Donoghue, Jr. *Monotone matrix functions and analytic continuation*, Springer–Verlag, New York–Heidelberg, 1974.
11. L. B. Golinskii. *A generalization of the matrix Nevanlinna–Pick problem*, Izv. Akad. Nauk Armyan. SSR Ser. Mat. **18** (1983), 187–205. (Russian).
12. A. Hindmarsh. *Pick conditions and analyticity*, Pacific J. Math. **27** (1968), 527–531.
13. M. G. Krein and H. Langer. *Über die verallgemeinerten Resolventen und die charakteristische Funktion eines isometrischen Operators im Raum  $\Pi_\kappa$* , Colloq. Math. Soc. János Bolyai **5** (1972), 353–399.

14. P. Lancaster and M. Tismenetsky. *The theory of matrices with applications*, 2nd Edition, Academic Press, 1985.
15. A. A. Nudelman. *A generalization of classical interpolation problems*, Dokl. Akad. Nauk. SSSR **4** (1981), 790–793. (Russian).

DEPARTMENT OF MATHEMATICS, COLLEGE OF WILLIAM AND MARY, WILLIAMSBURG, VA  
23187-8795

*E-mail address:* vlad@math.wm.edu, sykhei@wm.edu, lxrodm@math.wm.edu

*This page intentionally left blank*

# One-dimensional perturbations of selfadjoint operators with finite or discrete spectrum

Yu.M. Arlinskiĭ, S. Hassi, H.S.V. de Snoo, and E.R. Tsekanovskii

**ABSTRACT.** There are many classical results concerning the spectrum of perturbations of selfadjoint operators; in particular the behaviour under “small perturbations” has been investigated throughout. In the present paper attention is paid to the asymptotic behaviour of eigenvalues under rank one perturbations when such perturbations become infinitely large. This leads in a natural way to the study of the eigenvalues of a selfadjoint limiting perturbation, which is necessarily a selfadjoint relation (multivalued linear operator). The use of extension theory and associated  $Q$ -functions facilitates the study of the asymptotic properties under such “large perturbations” and leads to similar interpretations as are known in the case of “small perturbations”.

## 1. Introduction

Let the linear space  $\mathfrak{H} = \mathbb{C}^n$  be provided with the usual inner product denoted by  $(\cdot, \cdot)$ , let  $A$  be a selfadjoint operator in  $\mathfrak{H}$ , and let  $\omega \in \mathbb{C}^n$  be a nontrivial vector. A family of one-dimensional perturbations of  $A$  is defined as

$$(1.1) \quad A(\tau) = A + \tau(\cdot, \omega)\omega, \quad \tau \in \mathbb{R}.$$

Since  $\tau \in \mathbb{R}$ , the righthand side of (1.1) defines a selfadjoint operator in  $\mathfrak{H}$ . Many facts concerning such perturbations in finite-dimensional spaces can be found in [14], [26, Chapter II, §1–§2], [31], [34]. Denote the eigenvalues of  $A(\tau)$  in (1.1) by

$$\lambda_1(\tau) \leq \lambda_2(\tau) \leq \cdots \leq \lambda_n(\tau).$$

The study of the behaviour of these eigenvalues as  $\tau \rightarrow \pm\infty$  requires an interpretation of (1.1) for  $\tau = \pm\infty$ . It turns out that this limit is not an operator anymore, but that it is a relation (multivalued operator); this relation has  $\pm\infty$  as an eigenvalue with  $\omega$  as the corresponding eigenelement. This interpretation is not immediately clear from (1.1); in fact, it is only via Krein’s formula (see (2.2))

---

1991 *Mathematics Subject Classification.* Primary 47A70, 47B15, 47B25; Secondary 47A55, 47A57.

*Key words and phrases.* Rank one perturbation, symmetric operator, selfadjoint extension, Friedrichs extension, Krein-von Neumann extension.

The first author was supported in part by the Academy of Finland, project 52532.

The second author was supported in part by the Academy of Finland, project 40362.

The authors thank C.T. de Jong for his assistance in the preparation of this paper.

below) or an equivalent linear fractional transformation formula (see (7.1) and (7.2) below), that the structure of the limit becomes clear.

The purpose of this paper is to provide some additional, partially expository, information concerning one-dimensional perturbations of selfadjoint operators with discrete spectrum; and in particular of selfadjoint operators in finite-dimensional Hilbert spaces. It provides the above interpretation in a natural way: the arguments are completely elementary and can be seen as a natural completion of the discussion by W.F. Donoghue [13]. The present situation is closely related to the notion of coupling at infinity, which has been studied by F. Gesztesy, A. Kiselev, and B. Simon, cf. [16], [27], [32], for the infinite-dimensional case, see also [21], [22], [24], [30]. In addition, it is shown how this example fits in the extension theory of symmetric operators. In particular, the corresponding Friedrichs and Kreĭn-von Neumann extensions are determined, cf. [28]. A necessary and sufficient condition is given so that these extensions coincide. A matrix representation of  $A(\tau)$  with  $\tau \in \mathbb{C}$  in (1.1) with respect to the orthogonal decomposition  $\mathfrak{H} = (\text{span } \{\omega\})^\perp \oplus \text{span } \{\omega\}$  provides asymptotic results when  $\tau \in \mathbb{C}$  and  $\tau \rightarrow \infty$ . Finally some elementary examples of rank one perturbations of selfadjoint operators in infinite-dimensional Hilbert spaces are included. More general facts will be presented elsewhere.

## 2. Kreĭn's formula

The resolvent operator  $(A(\tau) - z)^{-1}$ ,  $z \in \mathbb{C} \setminus \mathbb{R}$ , of  $A(\tau)$  in (1.1) can be expressed as a perturbation of the resolvent operator  $(A - z)^{-1}$  by means of the so-called Kreĭn's formula. Define the scalar function  $Q(z)$  and the vector function  $\chi(z)$  by

$$(2.1) \quad Q(z) = ((A - z)^{-1}\omega, \omega), \quad \chi(z) = (A - z)^{-1}\omega.$$

These functions are rational and their poles are at the eigenvalues  $\lambda_i = \lambda_i(0)$ ,  $i = 1, 2, \dots, n$ , of  $A$ . The following result goes back to M.G. Kreĭn, cf. [1], [17]; it was mentioned in the finite-dimensional situation by W.F. Donoghue [13]. A very simple proof is included for completeness.

**PROPOSITION 2.1.** *For each  $\tau \in \mathbb{R}$  the resolvent operator of  $A(\tau)$  in (1.1) is given by*

$$(2.2) \quad (A(\tau) - z)^{-1} = (A - z)^{-1} - \chi(z) \frac{1}{Q(z) + 1/\tau} (\cdot, \chi(\bar{z})), \quad z \in \mathbb{C} \setminus \mathbb{R}.$$

**PROOF.** It follows from (1.1) that

$$(2.3) \quad \begin{aligned} (A(\tau) - z)^{-1} - (A - z)^{-1} &= (A(\tau) - z)^{-1}(A - A(\tau))(A - z)^{-1} \\ &= -\tau((A - z)^{-1}\cdot, \omega)(A(\tau) - z)^{-1}\omega. \end{aligned}$$

Apply (2.3) to the element  $\omega$  and solve  $(A(\tau) - z)^{-1}\omega$ :

$$(A(\tau) - z)^{-1}\omega = \frac{1}{1 + \tau Q(z)} (A - z)^{-1}\omega.$$

Kreĭn's formula follows when this result is substituted back in (2.3).  $\square$

The definitions (2.1) imply that  $\overline{Q(z)} = Q(\bar{z})$  and that

$$(2.4) \quad \frac{Q(z) - \overline{Q(w)}}{z - \bar{w}} = (\chi(z), \chi(w)).$$

Note that (2.4) is a direct consequence of the resolvent identity

$$(A - z)^{-1} - (A - w)^{-1} = (z - w)(A - z)^{-1}(A - w)^{-1}.$$

The rational function  $Q(z)$  is a Nevanlinna or Herglotz-Nevanlinna function, cf. [6], [25]; i.e., it is holomorphic on  $\mathbb{C} \setminus \mathbb{R}$ , it is symmetric with respect to  $\mathbb{R}$ , and it maps the upper halfplane  $\mathbb{C}^+$  into itself.

Clearly, the righthand side of (2.2) still makes sense when  $\tau \rightarrow \pm\infty$ . Hence, the formulation of the one-dimensional perturbations  $A(\tau)$  in (1.1) via Kreĭn's formula (2.2) makes it possible to consider their limiting behaviour as  $\tau \rightarrow \pm\infty$ .

### 3. Reduction of the problem

The operators  $A(\tau)$ ,  $\tau \in \mathbb{R}$ , in (1.1) have the same action as the operator  $A$ , when restricted to the orthogonal complement of the element  $\omega$ . Define therefore the domain restriction  $S$  of  $A$  to the orthogonal complement of the element  $\omega$ :

$$(3.1) \quad \text{dom } S = (\text{span } \{\omega\})^\perp.$$

Then  $S$  is a nondensely defined symmetric operator in  $\mathfrak{H}$  with defect spaces

$$\text{ran } (S - \bar{z})^\perp = \text{span } \{\chi(z)\}, \quad z \in \mathbb{C} \setminus \mathbb{R},$$

whose dimension equals one, i.e., the defect numbers of  $S$  in  $\mathfrak{H}$  are  $(1, 1)$ . Each one-dimensional perturbation  $A(\tau)$  in (1.1) is a selfadjoint operator extension of  $S$ . Let  $\mathfrak{H}_r$  be the linear span of all the eigenspaces of  $S$ , i.e., the linear span of all the eigenspaces of  $A$  which are orthogonal to  $\omega$ . The space  $\mathfrak{H}_r$  is an invariant subspace for  $A$ ; denote the restriction of  $A$  to  $\mathfrak{H}_r$  by  $A_r$ . Observe that  $A(\tau)h = Ah$ ,  $\tau \in \mathbb{R}$ , for each  $h \in \mathfrak{H}_r$ . Let  $\mathfrak{H}_s$  be the orthogonal complement of  $\mathfrak{H}_r$  and observe that  $\omega \in \mathfrak{H}_s$ . The space  $\mathfrak{H}_s$  is an invariant subspace for  $A$ ; denote the restriction of  $A$  to  $\mathfrak{H}_s$  by  $A_s$ . Let the operator  $S_s$  be the domain restriction of  $A_s$  to the orthogonal complement  $\mathfrak{H}_s \ominus \text{span } \{\omega\}$ :

$$(3.2) \quad \text{dom } S_s = \mathfrak{H}_s \ominus \text{span } \{\omega\}.$$

Then  $S_s$  is a nondensely defined symmetric operator in  $\mathfrak{H}_s$  with

$$\text{ran } (S_s - \bar{z})^\perp = \text{span } \{(A_s - z)^{-1}\omega\}, \quad z \in \mathbb{C} \setminus \mathbb{R},$$

i.e., the defect numbers of  $S_s$  in  $\mathfrak{H}_s$  are  $(1, 1)$ . The decomposition  $\mathfrak{H} = \mathfrak{H}_s \oplus \mathfrak{H}_r$  leads to a corresponding decomposition of the operators  $S$  and  $A$ :

$$S = S_s \oplus A_r, \quad A = A_s \oplus A_r,$$

and, clearly, the space  $\mathfrak{H}_s$  is invariant under the one-dimensional perturbations  $A(\tau)$ ,  $\tau \in \mathbb{R}$ . Observe that

$$Q(z) = ((A_s - z)^{-1}\omega, \omega), \quad z \in \mathbb{C} \setminus \mathbb{R},$$

and Kreĭn's formula may also be restricted to the space  $\mathfrak{H}_s$ .

The symmetric operator  $S_s$  is called the simple (or completely nonselfadjoint) part of the symmetric operator  $S$ . The operator  $S$  is called simple if  $S = S_s$ , or equivalently if  $\mathfrak{H}_r = \{0\}$ . In general, the operator  $S$  need not be simple. For instance, if  $\omega$  is an eigenvector of  $A$ , then the orthogonal complement  $(\text{span } \{\omega\})^\perp$  is invariant under  $A$  (and also under  $A(\tau)$ ,  $\tau \in \mathbb{R}$ ). Moreover, if the eigenvector  $\omega$  corresponds to the eigenvalue  $\tau_0$ , then it is also an eigenvector of  $A(\tau)$  corresponding

to the eigenvalue  $\lambda(\tau) = \tau_0 + \tau\|\omega\|^2$ . Hence,  $\lambda(\tau)/\tau \rightarrow \|\omega\|^2$ , and in particular,  $\tau_0 + \tau\|\omega\|^2 \rightarrow \pm\infty$  as  $\tau \rightarrow \pm\infty$ .

**PROPOSITION 3.1.** *The operator  $S$  is simple if and only if no eigenvector of  $A$  is orthogonal to  $\omega$ . In this case, the spectrum of  $A$  is necessarily simple, i.e. each eigenvalue of  $A$  has multiplicity 1.*

**PROOF.** The existence of an invariant subspace of  $A$  in  $\mathfrak{H} \ominus \text{span}\{\omega\}$  is equivalent to the existence of eigenvectors of  $A$  in  $\mathfrak{H} \ominus \text{span}\{\omega\}$ , which shows the first statement. Assume that  $\lambda_0$  is an eigenvalue with multiplicity 2, and let  $e_1$  and  $e_2$  be linearly independent eigenvectors. If  $(e_1, \omega) = c_1 \neq 0$  and  $(e_2, \omega) = c_2 \neq 0$ , then

$$e_1/c_1 - e_2/c_2 \neq 0$$

is an eigenvector belonging to the eigenvalue  $\lambda_0$ , such that  $(\omega, e_1/c_1 - e_2/c_2) = 0$ .  $\square$

Due to the above mentioned reduction with respect to the decomposition  $\mathfrak{H} = \mathfrak{H}_s \oplus \mathfrak{H}_r$ , it may be assumed without loss of generality that  $S$  is simple.

#### 4. The eigenvalues and their limiting behaviour

Let  $S$  be simple, so that the eigenvalues  $\lambda_i$ ,  $i = 1, 2, \dots, n$ , of the selfadjoint operator  $A$  have multiplicity 1:

$$\lambda_1 < \lambda_2 < \dots < \lambda_n.$$

Let  $e_i$ ,  $i = 1, 2, \dots, n$ , be the corresponding orthonormal eigenvectors and decompose  $\omega$  in terms of this basis:  $\omega = \sum_{i=1}^n (\omega, e_i)e_i$ . It follows from Proposition 3.1 that  $(\omega, e_i) \neq 0$  for  $i = 1, 2, \dots, n$ . With  $c_i = |(\omega, e_i)|^2$ , it follows that  $Q(z)$  in (2.1) has the representation in partial fractions

$$(4.1) \quad Q(z) = \frac{c_1}{\lambda_1 - z} + \frac{c_2}{\lambda_2 - z} + \dots + \frac{c_n}{\lambda_n - z},$$

where  $z \in \mathbb{C} \setminus \{\lambda_1, \lambda_2, \dots, \lambda_n\}$ . The constants  $c_i$ ,  $i = 1, \dots, n$ , are positive numbers with

$$(4.2) \quad c_1 + c_2 + \dots + c_n = \|\omega\|^2.$$

Observe that it follows from (2.1), or from (4.1) and (4.2), that

$$-zQ(z) \rightarrow \|\omega\|^2, \quad z \rightarrow \infty,$$

and that it follows from (4.1) that

$$(4.3) \quad Q'(z) = \frac{c_1}{(\lambda_1 - z)^2} + \frac{c_2}{(\lambda_2 - z)^2} + \dots + \frac{c_n}{(\lambda_n - z)^2}, \quad z \in \mathbb{C} \setminus \{\lambda_1, \lambda_2, \dots, \lambda_n\}.$$

The function  $Q(z)$  has therefore real zeros  $\mu_i$ ,  $i = 1, 2, \dots, n-1$ , which interlace with its poles as follows

$$\lambda_1 < \mu_1 < \lambda_2 < \mu_2 < \lambda_3 < \dots < \lambda_{n-1} < \mu_{n-1} < \lambda_n.$$

**PROPOSITION 4.1.** *Let  $S$  in (3.1) be simple. Then the eigenvalues  $\lambda_i(\tau)$ ,  $i = 1, 2, \dots, n$ , of  $A(\tau)$  in (1.1) coincide with the zeros of the function  $Q(z) + 1/\tau$ .*

**PROOF.** It follows from (4.1) that  $Q(z)$  is given by

$$(4.4) \quad \frac{c_1(\lambda_2 - z) \cdots (\lambda_n - z) + \dots + c_n(\lambda_1 - z) \cdots (\lambda_{n-1} - z)}{(\lambda_1 - z)(\lambda_2 - z) \cdots (\lambda_n - z)} = \|\omega\|^2 \frac{N(z)}{D(z)},$$

where the polynomials  $N(z)$  and  $D(z)$  are given by

$$(\mu_1 - z)(\mu_2 - z) \cdots (\mu_{n-1} - z), \quad (\lambda_1 - z)(\lambda_2 - z) \cdots (\lambda_n - z),$$

respectively. Since the space  $\mathfrak{H} = \mathbb{C}^n$  is finite-dimensional, the only singularities of the resolvent  $(A(\tau) - z)^{-1}$  are poles. It follows from Krein's formula (2.2) that these poles are located either at  $\lambda_i$ ,  $i = 1, 2, \dots, n$ , or at the zeros of the function  $Q(z) + 1/\tau$ . However, it is obvious that in Krein's formula (2.2) the poles of  $(A - z)^{-1}$  are compensated by the poles of

$$\chi(z) \frac{1}{Q(z) + 1/\tau} (\cdot, \chi(\bar{z})) = \chi(z) \frac{D(z)}{N(z) + D(z)/\tau} (\cdot, \chi(\bar{z})).$$

Hence, the eigenvalues  $\lambda_i(\tau)$ ,  $i = 1, 2, \dots, n$ , coincide with the zeros of the function  $Q(z) + 1/\tau$ .  $\square$

It follows from Proposition 4.1 that the eigenvalues interlace:

$$\lambda_1 < \lambda_1(\tau) < \lambda_2 < \lambda_2(\tau) < \cdots < \lambda_{n-1} < \lambda_{n-1}(\tau) < \lambda_n < \lambda_n(\tau), \quad \tau > 0,$$

and

$$\lambda_1(\tau) < \lambda_1 < \lambda_2(\tau) < \lambda_2 < \cdots < \lambda_{n-1}(\tau) < \lambda_{n-1} < \lambda_n(\tau) < \lambda_n, \quad \tau < 0,$$

since  $Q(z) \rightarrow \infty$  as  $z \rightarrow \infty$ . As the function  $Q(z)$  is holomorphic, the eigenvalues  $\lambda_i(\tau)$  depend holomorphically on the parameter  $\tau$ . If  $\tau$  increases or decreases, then the eigenvalues  $\lambda_i(\tau)$  increase or decrease, respectively, cf. (4.3). Moreover, if  $\tau \rightarrow \infty$  then  $\lambda_n(\tau)$  "escapes" to  $\infty$ , and if  $\tau \rightarrow -\infty$  then  $\lambda_1(\tau)$  "escapes" to  $-\infty$ . A direct consequence of (1.1) is the following trace formula

$$(4.5) \quad \lambda_1(\tau) + \lambda_2(\tau) + \cdots + \lambda_n(\tau) = \lambda_1 + \lambda_2 + \cdots + \lambda_n + \|\omega\|^2 \tau.$$

The following corollary sums up the above discussion.

**COROLLARY 4.2.** *Let  $S$  be simple. Then the asymptotic behaviour of the eigenvalues  $\lambda_i(\tau)$  as  $\tau \rightarrow \infty$ , and  $\lambda_{i+1}(\tau)$  as  $\tau \rightarrow -\infty$ , is given by*

$$\mu_i + o(1), \quad i = 1, 2, \dots, n - 1.$$

*The escaping eigenvalues  $\lambda_n(\tau)$  as  $\tau \rightarrow \infty$ , and  $\lambda_1(\tau)$  as  $\tau \rightarrow -\infty$ , have the asymptotic behaviour*

$$\|\omega\|^2 \tau + \sum_{i=1}^n \lambda_i - \sum_{i=1}^{n-1} \mu_i + o(1).$$

## 5. Interpretation of the limiting behaviour

In order to interpret the limiting values  $\mu_i$ ,  $i = 1, 2, \dots, n - 1$ , and  $\pm\infty$ , take the limit as  $\tau \rightarrow \pm\infty$  in the righthand side of Krein's formula (2.2):

$$(5.1) \quad R_\infty(z) = (A - z)^{-1} - \chi(z) \frac{1}{Q(z)} (\cdot, \chi(\bar{z})), \quad z \in \mathbb{C} \setminus \mathbb{R}.$$

Clearly,  $R_\infty(z)$  is a linear operator in  $\mathfrak{H}$ , and  $R_\infty(z)^* = R_\infty(\bar{z})$ ,  $z \in \mathbb{C} \setminus \mathbb{R}$ . Let  $P$  be the orthogonal projection from  $\mathfrak{H}$  onto  $\text{span}\{\omega\}$ . Then the operator  $(I - P)S$  is selfadjoint in  $\mathfrak{H} \ominus \text{span}\{\omega\}$ .

**PROPOSITION 5.1.** *Let the operator  $S$  in (3.1) be simple. The poles of the operator  $R_\infty(z)$  in (5.1) coincide with  $\mu_i$ ,  $i = 1, 2, \dots, n - 1$ . Moreover,  $\ker R_\infty(z) = \text{span}\{\omega\}$ , and in the orthogonal complement  $\mathfrak{H} \ominus \text{span}\{\omega\}$  the operator  $R_\infty(z)$  coincides with the resolvent of the selfadjoint operator  $(I - P)S$ .*

PROOF. The first statement, concerning the poles of  $R_\infty(z)$ , can be shown as in the proof of Proposition 4.1. Obviously,  $\omega \in \ker R_\infty(z)$ . Now let  $h \in \ker R_\infty(z)$ , then

$$(A - z)^{-1}h = \chi(z) \frac{1}{Q(z)} (h, \chi(\bar{z})),$$

and, by applying  $A - z$  to both sides of this equation it follows that

$$h = \omega \frac{1}{Q(z)} (h, \chi(\bar{z})),$$

which shows that  $h \in \text{span } \{\omega\}$ . The property  $R_\infty(z)^* = R_\infty(\bar{z})$  implies that  $R_\infty(z)$  maps into the orthogonal complement  $\mathfrak{H} \ominus \text{span } \{\omega\}$ . Hence it follows from (5.1) that

$$(S - z)R_\infty(z) = (A - z)R_\infty(z) = I - \omega \frac{1}{Q(z)} (\cdot, \chi(\bar{z})).$$

Therefore, also  $(I - P)(S - zI)R_\infty(z) = I$ .  $\square$

Hence the points  $\mu_i$ ,  $i = 1, 2, \dots, n - 1$ , are the eigenvalues of the compression  $(I - P)S$  of  $A$  to the orthogonal complement  $\mathfrak{H} \ominus \text{span } \{\omega\}$ . In fact, if

$$(5.2) \quad A(\infty) = (I - P)S \widehat{\oplus} (\{0\} \oplus \text{span } \{\omega\}).$$

where the sum  $\widehat{\oplus}$  is an orthogonal componentwise sum in the sense of graphs, then  $A(\infty)$  is a selfadjoint relation whose multivalued part

$$\text{mul } A(\infty) = \{ \varphi : \{0, \varphi\} \in A(\infty) \}$$

is spanned by  $\omega$ ; moreover,  $R_\infty(z) = (A(\infty) - z)^{-1}$  and  $\ker A(\infty) = \text{mul } A(\infty)$ . This multivalued part can be interpreted as the eigenspace corresponding to the “eigenvalue”  $\pm\infty$ . The relation  $A(\infty)$  is the only selfadjoint extension of  $S$  in (3.1) which is not an operator. According to (5.2) the operator  $(I - P)S$  is the “orthogonal operator part” of  $A(\infty)$ . However, since  $S$  is not densely defined, its Friedrichs extension is the only selfadjoint extension of  $S$  which is not an operator, in particular, it coincides with  $A(\infty)$ . To see this, recall that the Friedrichs extension of an operator (or more generally of a linear relation)  $S$  which is semibounded from below  $S \geq \alpha I$  can be defined by

$$(5.3) \quad S_F = \{ \{f, g\} \in S^* : f \in \text{dom } [S] \},$$

where  $S^*$  is the adjoint (linear relation) of  $S$  given by

$$S^* = \{ \{f, g\} \in \mathfrak{H}^2 : (g, h) = (f, k) \text{ for all } \{h, k\} \in S \}$$

and  $\text{dom } [S]$  stands for the completion of  $\text{dom } S$  with respect to the norm  $\|h\|_S^2 = \|h\|^2 + (k, h)$ ,  $\{h, k\} \in S$ , cf. [9], [10]. It follows from the definition of  $S$  in (3.1) that

$$(5.4) \quad S^* = A \widehat{\oplus} (\{0\} \oplus \text{span } \{\omega\}).$$

Moreover, since  $\mathfrak{H}$  is finite-dimensional the norms  $\|\cdot\|$  and  $\|\cdot\|_S$  are equivalent, so that  $\text{dom } [S] = \text{dom } S$ . Now, it is clear from (5.3) and (5.4) that  $S_F = A(\infty)$ .

Since  $S$  is bounded from below, its Friedrichs extension preserves the lower bound. In fact, the lower and upper bounds can be given precisely.

COROLLARY 5.2. *Let the operator  $S$  in (3.1) be simple. Then*

$$\inf \left\{ \frac{(Sf, f)}{(f, f)} : f \in \text{dom } S \right\} = \mu_1, \quad \sup \left\{ \frac{(Sf, f)}{(f, f)} : f \in \text{dom } S \right\} = \mu_{n-1}.$$

PROOF. Observe, for instance, that

$$\inf \left\{ \frac{(Sf, f)}{(f, f)} : f \in \text{dom } S \right\} = \inf \left\{ \frac{((I - P)Sf, f)}{(f, f)} : f \in \mathbb{C}^n \ominus \text{span } \{\omega\} \right\},$$

which is precisely the lower bound of the operator  $(I - P)S$ . The eigenvalues of  $(I - P)S$  are given by  $\mu_i$ ,  $i = 1, 2, \dots, n - 1$ , cf. Proposition 5.1.  $\square$

These results are special cases of general results concerning Herglotz-Nevanlinna functions with finite total mass, cf. [21], [22].

## 6. Further asymptotic behaviour

The asymptotic behaviour of the escaping eigenvalues in Corollary 4.2 matches the occurrence of the infinite eigenvalue  $\pm\infty$  of the Friedrichs extension  $A(\infty)$ . Further terms in the asymptotic expansion of the escaping eigenvalue can be obtained.

**PROPOSITION 6.1.** *Let  $S$  be simple. Then the asymptotic behaviour of the eigenvalues  $\lambda_i(\tau)$  as  $\tau \rightarrow \infty$ , and  $\lambda_{i+1}(\tau)$  as  $\tau \rightarrow -\infty$ , is given by*

$$(6.1) \quad \mu_i - \frac{1}{Q'(\mu_i)} \frac{1}{\tau} + o\left(\frac{1}{\tau}\right), \quad i = 1, 2, \dots, n - 1.$$

Moreover, the asymptotic behaviour of the escaping eigenvalues  $\lambda_n(\tau)$  as  $\tau \rightarrow \infty$ , and  $\lambda_1(\tau)$  as  $\tau \rightarrow -\infty$ , is given by

$$(6.2) \quad \|\omega\|^2 \tau + \sum_{i=1}^n \lambda_i - \sum_{i=1}^{n-1} \mu_i + \left( \sum_{i=1}^{n-1} \frac{1}{Q'(\mu_i)} \right) \frac{1}{\tau} + o\left(\frac{1}{\tau}\right).$$

PROOF. According to Proposition 4.1 the eigenvalues  $\lambda_i(\tau)$ ,  $i = 1, 2, \dots, n$ , of  $A(\tau)$  in (1.1) coincide with the zeros of the function  $Q(z) + 1/\tau$ . The asymptotic results now follow by implicit differentiation and an application of (4.5).  $\square$

More terms in the asymptotic expansion of the eigenvalues may be obtained in a similar way, cf. [12]. For instance, (6.1) can be refined:

$$\mu_i - \frac{1}{Q'(\mu_i)} \frac{1}{\tau} - \frac{Q''(\mu_i)}{2Q'(\mu_i)^3} \frac{1}{\tau^2} + o\left(\frac{1}{\tau^2}\right),$$

which also leads to more precise estimates for the escaping eigenvalues. In fact, with the polynomials  $N(z)$  and  $D(z)$  defined in (4.4) the equation for the perturbed eigenvalues becomes

$$\|\omega\|^2 N(z) + \frac{1}{\tau} D(z) = 0.$$

This is an algebraic equation and it follows from a well-known result in function theory that the roots of this equation are real analytic in  $1/\tau$  [or (branches of) analytic functions in  $1/\tau$  if  $\tau$  is a complex parameter]. As  $\tau \rightarrow \pm\infty$  there are  $n - 1$  finite roots  $\lambda_i(\tau)$  which tend to the zeros  $\mu_i$  of  $N(z)$  as  $\tau \rightarrow \pm\infty$ . The numbers  $\lambda_i(\tau)$  are eigenvalues of the selfadjoint operator  $A(\tau)$ ,  $\tau \in \mathbb{R}$ , while the numbers  $\mu_i$  are the eigenvalues of the selfadjoint operator  $(I - P)S$ , cf. Proposition 5.1. All of these eigenvalues have multiplicity 1, cf. Proposition 3.1 and Section 4. Therefore there is no branch point at  $1/\tau = 0$  and it follows that the roots  $\lambda_i(\tau)$  tending to  $\mu_i$ , cf. (6.1), have asymptotic expansions (Puiseux series) in  $1/\tau$  of the form

$$\lambda_i(\tau) = \mu_i + \sum_{k=1}^{\infty} \frac{a_k}{\tau^k},$$

cf. [26]. To treat the escaping eigenvalue write the rank one perturbation in a different way:

$$(6.3) \quad A(\tau) = A + \tau(\cdot, \omega)\omega = \tau \left( \frac{1}{\tau}A + (\cdot, \omega)\omega \right).$$

Consider in (6.3) the eigenvalues of  $(1/\tau)A + (\cdot, \omega)\omega$ . This sum can be interpreted as a perturbation of the one-dimensional operator  $(\cdot, \omega)\omega$  whose only nonzero eigenvalue is given by  $\|\omega\|^2$  which has multiplicity one. Now, arguments similar to those used above yield the following asymptotic expansion in  $1/\tau$  for the perturbed eigenvalue  $\|\omega\|^2$  of  $(1/\tau)A + (\cdot, \omega)\omega$ :

$$\|\omega\|^2 + \sum_{k=1}^{\infty} \frac{b_k}{\tau^k}.$$

In view of (6.3) multiplying this expansion by  $\tau$  one obtains the asymptotic expansion for the escaping eigenvalue, which parallels (6.2).

## 7. Extension theory

The one-dimensional perturbations (1.1) can be also considered from the point of view of  $Q$ -functions. The function  $Q(z)$  in (2.1) is the  $Q$ -function of  $A$  and its symmetric restriction  $S$ . Likewise the functions

$$(7.1) \quad Q_\tau(z) = \frac{Q(z) - \tau}{1 + \tau Q(z)}, \quad \tau \in \mathbb{R} \cup \{\infty\},$$

are the  $Q$ -functions of all the selfadjoint extensions  $A(\tau)$ ,  $\tau \in \mathbb{R} \cup \{\infty\}$ , and  $S$ . In fact, with  $\omega_\tau = \sqrt{\tau^2 + 1}\omega$ , and

$$Q_\tau(z) = -\tau + ((A(\tau) - z)^{-1}\omega_\tau, \omega_\tau), \quad \chi_\tau(z) = (A(\tau) - z)^{-1}\omega_\tau,$$

it follows that

$$\frac{Q_\tau(z) - \overline{Q_\tau(w)}}{z - \bar{w}} = (\chi_\tau(z), \chi_\tau(w)),$$

see also [15], [17], [21], [29]. The expressions also make sense for  $\tau \rightarrow \infty$ , when

$$(7.2) \quad Q_\infty(z) = -\frac{1}{Q(z)}, \quad \chi_\infty(z) = \frac{\chi(z)}{Q(z)},$$

so that

$$\frac{Q_\infty(z) - \overline{Q_\infty(w)}}{z - \bar{w}} = (\chi_\infty(z), \chi_\infty(w)).$$

This limit corresponds to taking the limit in Krejn's formula (2.2). It is clear that

$$\lim_{y \rightarrow \infty} \frac{\operatorname{Im} Q_\tau(iy)}{y} = 0, \quad \tau \in \mathbb{R}, \quad \lim_{y \rightarrow \infty} \frac{\operatorname{Im} Q_\infty(iy)}{y} = \frac{1}{\|\omega\|^2},$$

which expresses in analytical terms the fact that  $A(\infty)$  is the only selfadjoint extension of  $S$  which is not an operator.

Now consider the case that  $S$  is simple, so that  $Q(z)$  has the partial fraction expansion (4.1) with  $\lambda_1 < \lambda_2 < \dots < \lambda_n$  and  $c_i > 0$ ,  $i = 1, 2, \dots, n$ . Note that for any function  $Q(z)$  of the form (4.1) there is (up to isometric isomorphisms) an  $n$ -dimensional Hilbert space, a selfadjoint operator  $A$  in  $\mathfrak{H}$ , and an element  $\omega$ , such that (2.1) is a minimal representation for  $Q(z)$  (minimal in the sense that the corresponding symmetric operator is simple). In the case that  $S$  is simple the

functions  $Q_\tau(z) + \tau$ ,  $\tau \in \mathbb{R}$ , have similar partial fraction expansions (with  $c_i$  and  $\lambda_i$  replaced by  $c_i(\tau)$  and  $\lambda_i(\tau)$ ). However, for  $\tau = \infty$  the situation is slightly different:

$$Q_\infty(z) = \frac{z}{\|\omega\|^2} - \frac{\sum_{i=1}^n \lambda_i - \sum_{i=1}^{n-1} \mu_i}{\|\omega\|^2} + \frac{c_1(\infty)}{\mu_1 - z} + \cdots + \frac{c_{n-1}(\infty)}{\mu_{n-1} - z}, \quad z \in \mathbb{C} \setminus \mathbb{R},$$

where  $c_i(\infty) = 1/Q'(\mu_i)$ ,  $i = 1, 2, \dots, n-1$ . Like  $Q(z)$  each of the functions  $Q_\tau(z)$  determines (up to isometric isomorphisms) a minimal representation from  $A(\tau)$  and  $S$ , which can be read off from the corresponding partial fraction expansion, see [20].

Assume now that the lower bound  $\mu_1$  of  $S$  is nonnegative. In this case the nonnegative extensions  $A(\tau) \geq 0$  of  $S$  form an “operator interval” between the Friedrichs extension  $S_F$  and the so-called Kreĭn-von Neumann extension  $S_N$  in the sense that

$$(S_F + a)^{-1} \leq (A(\tau) + a)^{-1} \leq (S_N + a)^{-1}, \quad a > 0,$$

cf. [28]; see [19] for a simple presentation. Recall that for nonnegative  $S$  the Kreĭn-von Neumann extension  $S_N$  can be defined via

$$(7.3) \quad S_N = ((S^{-1})_F)^{-1},$$

where  $(S^{-1})_F$  denotes the Friedrichs extension of  $S^{-1}$ . Further general information concerning the Kreĭn-von Neumann extension can be found in [2], [3], [4], [5], [10], [11], [18].

**PROPOSITION 7.1.** *Let  $S$  be simple, and assume that the lower bound  $\mu_1$  of  $S$  is nonnegative. The Kreĭn-von Neumann extension of  $S$  corresponds to  $\tau = \tau_0$  with  $\tau_0 = -1/Q(0)$ . In particular:*

- (i) *If  $\mu_1 > 0$ , then  $\tau_0 < 0$  for  $\lambda_1 > 0$ ,  $\tau_0 = 0$  for  $\lambda_1 = 0$ , and  $\tau_0 > 0$  for  $\lambda_1 < 0$ .*
- (ii) *If  $\mu_1 = 0$ , then  $\tau_0 = \infty$ .*

*The extensions  $A(\tau)$  are nonnegative if and only if  $\tau_0 \leq \tau \leq \infty$ .*

**PROOF.** The value  $\tau_0$  corresponds to the situation where the smallest eigenvalue of  $A(\tau_0)$  is 0, i.e.,  $Q(0) = -1/\tau_0$ . Clearly, for  $\tau > \tau_0$  all eigenvalues of  $A(\tau)$  are positive. When  $\mu_1 > 0$ , then the three situations in (i) occur; with the interpretation that  $\tau_0 = 0$  when  $\lambda_1 = 0$  is a pole for  $Q(z)$ ). Likewise, when  $\mu_1 = 0$ , then  $\tau_0 = \infty$  since  $Q(0) = 0$ .  $\square$

If  $\mu_1 > 0$ , the Kreĭn-von Neumann extension is an operator given by  $A(\tau_0) = S \hat{+} (\ker S^* \oplus \{0\})$ , where  $\hat{+}$  denotes the componentwise sum (with  $S$  identified with its graph). If  $\mu_1 = 0$ , the Friedrichs extension  $A(\infty)$  coincides with the Kreĭn-von Neumann extension: it is the only nonnegative selfadjoint extension of  $S$ .

## 8. A matrix representation of the rank one perturbations

Let  $\mathfrak{H}_0 = \mathfrak{H} \ominus \text{span}\{\omega\}$  and let  $S_0 = (I - P)S$ . Then  $S_0$  is a selfadjoint operator in  $\mathfrak{H}_0$  and the operator  $A$  has the following block matrix representation with respect to the decomposition  $\mathfrak{H} = \mathfrak{H}_0 \oplus \text{span}\{\omega\}$ :

$$(8.1) \quad A = \begin{pmatrix} S_0 & B \\ B^* & D \end{pmatrix},$$

where

$$B = (I - P)A \upharpoonright \text{span}\{\omega\}, \quad B^* = PA \upharpoonright \mathfrak{H}_0, \quad D = \|\omega\|^{-2} (A\omega, \omega).$$

It follows from the trace formula applied to (8.1) that

$$\sum_{i=1}^n \lambda_i - \sum_{i=1}^{n-1} \mu_i = \|\omega\|^{-2} (A\omega, \omega).$$

For each  $\tau \in \mathbb{C}$  the operator  $A(\tau)$  given by (1.1) has the matrix representation

$$(8.2) \quad A(\tau) = \begin{pmatrix} S_0 & B \\ B^* & D + \tau \|\omega\|^2 \end{pmatrix}.$$

Then the operator  $A(\tau)$  is dissipative ( $\operatorname{Im} A(\tau) \geq 0$ ) when  $\operatorname{Im} \tau > 0$  and antidiSSIPATIVE ( $\operatorname{Im} A(\tau) \leq 0$ ) when  $\operatorname{Im} \tau < 0$ . Let the function  $a_\tau(z)$  be the Schur complement of  $A(\tau) - zI$ :

$$a_\tau(z) = \|\omega\|^{-2} (A\omega, \omega) + \tau \|\omega\|^2 - B^* (S_0 - zI)^{-1} B - z.$$

According to the Frobenius formula [23]

$$\det(A(\tau) - zI) = a_\tau(z) \det(S_0 - zI), \quad z \in \mathbb{C} \setminus \{\mu_1, \mu_2, \dots, \mu_{n-1}\},$$

so that the eigenvalues of  $A(\tau)$  are zeros of the rational function  $a_\tau(z)$ . Moreover,

(8.3)

$$(A(\tau) - zI)^{-1} = \begin{pmatrix} (S_0 - zI)^{-1} + \frac{1}{a_\tau(z)} (S_0 - zI)^{-1} BB^* (S_0 - zI)^{-1} & -\frac{1}{a_\tau(z)} (S_0 - zI)^{-1} B \\ -\frac{1}{a_\tau(z)} B^* (S_0 - zI)^{-1} & \frac{1}{a_\tau(z)} \end{pmatrix}.$$

which leads to

$$\|\omega\|^{-2} ((A(\tau) - zI)^{-1} \omega, \omega) = \frac{1}{a_\tau(z)}.$$

It follows from (7.1) that

$$(8.4) \quad a_\tau(z) = \frac{1 + \tau Q(z)}{Q(z)}.$$

**PROPOSITION 8.1.** *The asymptotic behaviour of  $(A(\tau) - zI)^{-1}$  is given by*

$$\lim_{\tau \rightarrow \infty} (A(\tau) - zI)^{-1} = \begin{pmatrix} (S_0 - zI)^{-1} & 0 \\ 0 & 0 \end{pmatrix}, \quad z \in \mathbb{C} \setminus \{\mu_1, \mu_2, \dots, \mu_{n-1}\}.$$

**PROOF.** It follows from (8.4) that  $a_\tau(z) \rightarrow \infty$  as  $\tau \rightarrow \infty$  for each  $z \in \mathbb{C}$  for which  $Q(z) \neq 0$ . The result then follows from (8.3).  $\square$

According to (8.4) the eigenvalues of  $A(\tau)$  are roots of the algebraic equation  $\tau Q(z) + 1 = 0$ . For  $\tau \in \mathbb{C} \setminus \mathbb{R}$  this equation can have solutions  $z$  with multiplicities larger than 1; in which case these solutions also satisfy the equation  $Q'(z) = 0$ . By (4.3) this equation has an even number of nonreal different solutions  $z_1, \bar{z}_1, \dots, z_m, \bar{z}_m$ ,  $m \leq n-1$ . The corresponding exceptional values of  $\tau$  are

$$\tau_k = -\frac{1}{Q(z_k)}, \quad \bar{\tau}_k = -\frac{1}{Q(\bar{z}_k)}, \quad k = 1, \dots, m.$$

The behaviour of the complex eigenvalues  $\lambda(\tau)$  of  $A(\tau)$  for  $\tau \rightarrow \infty$  is the same as for real  $\tau$ . Actually, by (4.4) and according Rouché's Theorem the equation

$$Q(z) = -\frac{1}{\tau}$$

has the unique solution  $\lambda_k(\tau)$  if  $|\tau| > R$  in any sufficiently small neighborhood of  $\mu_k$  for every  $k = 1, \dots, n - 1$ , and in any neighborhood of infinity  $|z| > r$  for sufficiently large  $r$ . Let  $\tau$  be a complex number with sufficiently large module and choose the eigenvalues  $\lambda_i(\tau)$ ,  $i = 0, 1, \dots, n - 1$ , such that

$$(8.5) \quad \lim_{\tau \rightarrow \infty} \lambda_0(\tau) = \infty, \quad \lim_{\tau \rightarrow \infty} \lambda_k(\tau) = \mu_k, \quad k = 1, 2, \dots, n - 1.$$

**PROPOSITION 8.2.** *The asymptotic behaviour of the eigenvalue  $\lambda_0(\tau)$  is given by*

$$\lambda_0(\tau) = \tau \|\omega\|^2 + \|\omega\|^{-2} (A\omega, \omega) + o(1), \quad \tau \rightarrow \infty.$$

**PROOF.** Apply the trace formula to (8.2), so that

$$\sum_{i=0}^{n-1} \lambda_i(\tau) = \sum_{i=1}^{n-1} \mu_i + \|\omega\|^{-2} (A\omega, \omega) + \tau \|\omega\|^2.$$

The result then follows from (8.5).  $\square$

## 9. Final remarks

Let  $A$  be a selfadjoint operator in an infinite-dimensional Hilbert space  $\mathfrak{H}$  and let  $\omega \in \mathfrak{H}$  be nontrivial. Let the open interval  $\Gamma \subset \mathbb{R}$  be a gap in the spectrum of  $A$ . The perturbed operator  $A(\tau)$  in (1.1) has an eigenvalue depending on  $\tau$  in  $\Gamma$  which behaves as in the finite-dimensional case, whether the gap is bounded or unbounded. In particular, if  $A$  has a compact resolvent, such a situation prevails. If, in the general case, the operator  $A$  is not semibounded, then there does not exist an eigenvalue that escapes to  $\pm\infty$ , although the “limit”  $A(\infty)$  has an eigenvalue  $\infty$ . Below some illustrations involving differential operators are given, cf. [7], [8], [33]. In these examples the precise behaviour of the solutions of the eigenvalue equations (9.1), (9.4), and (9.7), follows from the series expansions of the trigonometric functions cot and tan (involving the Bernoulli numbers).

**EXAMPLE 9.1.** Let  $A$  be the selfadjoint operator in  $\mathfrak{H} = L^2(0, 1)$  generated by the differential operator  $iD := i\frac{d}{dx}$  with maximal domain and the selfadjoint boundary conditions  $u(0) = u(1)$ , and let  $\omega(x) = x$ . The eigenvalues of the unperturbed operator are given by  $e^{-i\lambda} = 1$ ; hence,  $\lambda = k2\pi$ ,  $k \in \mathbb{Z}$ . The eigenvalues of  $A(\tau)$ ,  $\tau \in \mathbb{R}$ , are given by the roots of

$$(9.1) \quad \frac{1}{2} \cot \frac{\lambda}{2} = \frac{1}{\lambda} - \frac{\lambda}{3} + \frac{\lambda^2}{\tau}, \quad \lambda \in \mathbb{R} \setminus \{0\}.$$

Hence, the original eigenvalues move to the right as  $\tau > 0$  increases, and they move to the left as  $\tau < 0$  decreases. The limits as  $\tau \rightarrow \pm\infty$  of these eigenvalues are given by the roots of

$$(9.2) \quad \frac{1}{2} \cot \frac{\lambda}{2} = \frac{1}{\lambda} - \frac{\lambda}{3}, \quad \lambda \in \mathbb{R} \setminus \{0\},$$

which correspond to the eigenvalues of the boundary-value problem

$$(9.3) \quad (I - P)(iD)u = \lambda u, \quad u(0) = u(1), \quad u \in \mathfrak{H} \ominus \text{span}\{\omega\},$$

where  $P$  stands for the orthogonal projection from  $L^2(0, 1)$  onto  $\text{span}\{\omega\}$ . The boundary-value problem (9.3) corresponding to the limiting eigenvalue equation can be interpreted as an eigenvalue problem for the compression of the selfadjoint

operator  $A$  to the closed linear subspace  $(\text{span}\{\omega\})^\perp$ . In this interpretation the element  $\omega$  is an eigenfunction corresponding to the eigenvalue  $\infty$  of the “limit”  $A(\infty)$  of the rank one perturbations  $A(\tau)$  as  $\tau \rightarrow \pm\infty$ . Observe that the corresponding  $Q$ -function  $Q(z) = ((A - z)^{-1}\omega, \omega)$  is given by

$$Q(z) = -\frac{1}{2z^2} \cot \frac{z}{2} + \frac{1}{z^3} - \frac{1}{3z}, \quad z \in \mathbb{C} \setminus \mathbb{R}.$$

**EXAMPLE 9.2.** Now let  $A$  be the selfadjoint operator in  $\mathfrak{H} = L^2(0, 1)$  generated by  $iD$  and the selfadjoint boundary conditions  $u(0) = u(1)$  as in Example 9.1, but let  $\omega(x) = 1$ . Then  $\omega$  is an eigenfunction of  $A$  corresponding to the eigenvalue 0. Hence, the underlying operator  $S$  is not simple anymore, and the Hilbert space  $\mathfrak{H} = L^2(0, 1)$  splits as  $\text{span}\{\omega\}$  and its orthogonal complement. In the orthogonal complement  $(\text{span}\{\omega\})^\perp$  the perturbation reduces to the differential operator  $iD$  with the given boundary conditions, so that its eigenvalues are given by  $k2\pi$ ,  $k \in \mathbb{Z} \setminus \{0\}$ . In  $\text{span}\{\omega\}$  the original operator  $A$  is the zero operator and the perturbation  $A(\tau)$  means multiplication by  $\tau$ . The corresponding eigenvalue of  $A(\tau)$  escapes to  $\pm\infty$  as  $\tau \rightarrow \pm\infty$ , and the compression of  $A$  in  $\text{span}\{\omega\}$  to the orthogonal complement of  $\omega$  reduces to the zero operator, defined on  $\{0\}$ . Again,  $\omega$  is an eigenelement corresponding to the eigenvalue  $\infty$ , which is reached in a continuous way as  $\tau \rightarrow \pm\infty$ . The  $Q$ -function generated by  $A$  and  $\omega$  in  $\text{span}\{\omega\}$  is given by

$$Q(z) = -\frac{1}{z}, \quad z \in \mathbb{C} \setminus \mathbb{R}.$$

**EXAMPLE 9.3.** Let  $A$  be the selfadjoint operator in  $\mathfrak{H} = L^2(0, 1)$  generated by  $-D^2$  and the selfadjoint boundary conditions  $u(0) = u(1) = 0$ , and let  $\omega(x) = x$ . The eigenvalues of the unperturbed operator are given by  $\sin \sqrt{\lambda} = 0$ ,  $\lambda \geq 0$ ; hence,  $\lambda = k^2\pi^2$ ,  $k \in \mathbb{N}$ . All, but at most one, of the eigenvalues of  $A(\tau)$ ,  $\tau \in \mathbb{R}$ , are positive and given by the roots of

$$(9.4) \quad \cot \sqrt{\lambda} = \frac{1}{\sqrt{\lambda}} - \frac{\sqrt{\lambda}}{3} + \frac{\lambda \sqrt{\lambda}}{\tau}, \quad \lambda > 0.$$

The original eigenvalues move to the right as  $\tau > 0$  increases and have a finite limit. They move to the left as  $\tau < 0$  decreases and have a finite limit, except the first eigenvalue and that one escapes to  $-\infty$ . The finite limits are given by the roots of

$$(9.5) \quad \cot \sqrt{\lambda} = \frac{1}{\sqrt{\lambda}} - \frac{\sqrt{\lambda}}{3}, \quad \lambda > 0,$$

which correspond to the eigenvalues of the boundary-value problem

$$(9.6) \quad (I - P)(-D^2)u = \lambda u, \quad u(0) = u(1) = 0, \quad u \in \mathfrak{H} \ominus \text{span}\{\omega\},$$

where  $P$  stands for the orthogonal projection from  $L^2(0, 1)$  onto  $\text{span}\{\omega\}$ . Actually, it is only for  $\tau > -45$ , that all eigenvalues of the perturbed problem are positive. For  $\tau = -45$  the smallest eigenvalue of the perturbed problem is zero, and for  $\tau < -45$ , all but the smallest eigenvalue are positive. The smallest eigenvalue  $\lambda_0(\tau)$  is negative. Clearly,  $\lambda_0(\tau)$  escapes to  $-\infty$  with  $\lambda_0(\tau)/\tau \rightarrow 1/3$  as  $\tau \rightarrow -\infty$ . As in Example 9.1, the boundary-value problem (9.6) corresponding to the limiting eigenvalue equation can be interpreted as an eigenvalue problem for the compression of the selfadjoint operator  $A$  to the closed linear subspace  $(\text{span}\{\omega\})^\perp$ . In this interpretation the element  $\omega$  is an eigenfunction corresponding to the eigenvalue  $\infty$

of the “limit”  $A(\infty)$  of the rank one perturbations  $A(\tau)$  as  $\tau \rightarrow \infty$ . The  $Q$ -function generated by  $A$  and  $\omega$  is given by

$$Q(z) = -\frac{1}{z\sqrt{z}} \cot \sqrt{z} + \frac{1}{z^2} - \frac{1}{3z}, \quad z \in \mathbb{C} \setminus \mathbb{R}.$$

**EXAMPLE 9.4.** Now let  $A$  be the selfadjoint operator in  $\mathfrak{H} = L^2(0, 1)$  generated by  $-D^2$  and the selfadjoint boundary conditions  $u(0) = u(1) = 0$  as in Example 9.3, but let  $\omega(x) = 1$ . The eigenvalues of the unperturbed problem are given by  $\sin \sqrt{\lambda} = 0, \lambda > 0$ ; hence,  $\lambda = k^2\pi^2, k \in \mathbb{N}$ . All, but at most one, of the eigenvalues of  $A(\tau)$  are positive and given by the roots of each of the following equations

$$\sin \frac{\sqrt{\lambda}}{2} = 0, \quad 2 \tan \frac{\sqrt{\lambda}}{2} = \left(1 - \frac{\lambda}{\tau}\right) \sqrt{\lambda}, \quad \lambda > 0.$$

Each root of the equation  $\sin \frac{\sqrt{\lambda}}{2} = 0, \lambda > 0$ , is an eigenvalue with eigenfunction  $\sin \sqrt{\lambda}x$ , which is orthogonal to  $\omega$ . Hence the eigenvalues  $4k^2\pi^2, k \in \mathbb{N}$ , remain at the same place for  $\tau \neq 0$ . The other original eigenvalues  $(2k+1)^2\pi^2, k \in \mathbb{N}$ , move to the right as  $\tau > 0$  increases and have a finite limit. They move to the left as  $\tau < 0$  decreases and have a finite limit, except the smallest eigenvalue which escapes to  $-\infty$  when  $\tau \rightarrow -\infty$ . The finite limits are given by the roots of

$$(9.7) \quad 2 \tan \frac{\sqrt{\lambda}}{2} = \sqrt{\lambda}, \quad \lambda > 0,$$

which correspond to the eigenvalues of the boundary-value problem

$$(9.8) \quad (I - P)(-D^2)u = \lambda u, \quad u(0) = u(1) = 0, \quad u \in \mathfrak{H} \ominus \text{span}\{\omega\},$$

where  $P$  stands for the orthogonal projection from  $L^2(0, 1)$  onto  $\text{span}\{\omega\}$ . The eigenvalues of the perturbed problem are all positive when  $\tau > -12$ . For  $\tau = -12$  the smallest eigenvalue of the perturbed problem is zero and for  $\tau < -12$ , all but the smallest eigenvalues are positive. The smallest eigenvalue  $\lambda_0(\tau)$  escapes to  $-\infty$  with  $\lambda_0(\tau)/\tau \rightarrow 1$  as  $\tau \rightarrow -\infty$ . Obviously, the underlying operator  $S$  is not simple anymore. In the Hilbert space  $\overline{\text{span}}\{\sin 2k\pi x : k \in \mathbb{N}\}$  the perturbation reduces to the differential operator  $-D^2$  with the given boundary conditions. In the orthogonal complement  $\mathfrak{H} \ominus \overline{\text{span}}\{\sin 2k\pi x : k \in \mathbb{N}\}$ , the eigenvalue equation corresponding to the perturbation is given by

$$(9.9) \quad 2 \tan \frac{\sqrt{\lambda}}{2} = \left(1 - \frac{\lambda}{\tau}\right) \sqrt{\lambda}, \quad \lambda > 0.$$

Its limiting form for  $\tau = \infty$  is the eigenvalue problem (9.8) for the compression to  $(\text{span}\{\omega\})^\perp$  of the selfadjoint operator  $A$  in  $\mathfrak{H} \ominus \overline{\text{span}}\{\sin 2k\pi x : k \in \mathbb{N}\}$ . The  $Q$ -function generated by  $A$  and  $\omega$  is given by

$$Q(z) = \frac{2}{z\sqrt{z}} \tan \frac{\sqrt{z}}{2} - \frac{1}{z}, \quad z \in \mathbb{C} \setminus \mathbb{R}.$$

## References

- [1] N.I. Achieser and I.M. Glasmann, *Theorie der linearen Operatoren im Hilbertraum*, 8. Auflage, Akademie Verlag, Berlin, 1981.
- [2] A. Alonso and B. Simon, "The Birman-Krein-Vishik theory of self-adjoint extensions of semi-bounded operators", J. Operator Theory, 4 (1980), 251–270.
- [3] T. Ando and K. Nishio, "Positive selfadjoint extensions of positive symmetric operators", Tôhoku Math. J., 22 (1970), 65–75.

- [4] Yu.M. Arlinskiĭ, S. Hassi, Z. Sebestyén, and H.S.V. de Snoo, "On the class of extremal extensions of a nonnegative operator", Oper. Theory Adv. Appl., 127 (2001), 41–81.
- [5] Yu. M. Arlinskiĭ and E. R. Tsekanovskii, "On von Neumann's parametrization of nonnegative selfadjoint extensions", preprint, 2001.
- [6] F.V. Atkinson, *Discrete and continuous boundary problems*. Academic Press, New York, 1968.
- [7] E.A. Catchespole, "A Cauchy problem for an ordinary integro-differential equation", Proc. Roy. Soc. Edinburgh (A), 72 (1972–73), 39–55.
- [8] E.A. Catchespole, "An integro-differential operator", J. London Math. Soc., 6 (1973), 513–523.
- [9] E.A. Coddington, "Extension theory of formally normal and symmetric subspaces", Mem. Amer. Math. Soc., 134 (1973).
- [10] E.A. Coddington and H.S.V. de Snoo, "Positive selfadjoint extensions of positive symmetric subspaces", Math. Z., 159 (1978), 203–214.
- [11] V.A. Derkach, S. Hassi, M.M. Malamud, and H.S.V. de Snoo, "Weyl functions and intermediate extensions of symmetric operators", Reports Nat. Acad. Sci. Ukraine (Dopov. Nats. Akad. Nauk Ukrainskoi), 10 (2001), 33–39.
- [12] V.A. Derkach, S. Hassi, and H.S.V. de Snoo, "Rank one perturbations in a Pontryagin space with one negative square", J. Funct. Anal., 188 (2002), 317–349.
- [13] W.F. Donoghue, *Monotone matrix functions and analytic continuation*, Springer-Verlag, Berlin-Heidelberg-New York, 1974.
- [14] K. Friedrichs, *Perturbation of spectra in Hilbert space*, American Mathematical Society, Providence, Rhode Island, 1965.
- [15] F. Gesztesy, N. Kalton, K. Makarov, and E.R. Tsekanovskii, "Some applications of operator-valued Herglotz functions", Oper. Theory Adv. Appl., 123 (2001), 272–321.
- [16] F. Gesztesy and B. Simon, "Rank one perturbations at infinite coupling", J. Funct. Anal., 128 (1995), 245–252.
- [17] F. Gesztesy and E.R. Tsekanovskii, "On matrix-valued Herglotz functions", Math. Nachr., 218 (2000), 61–138.
- [18] S. Hassi, M. Kaltenbäck, and H.S.V. de Snoo, "Generalized Krein-von Neumann extensions and associated operator models", Acta Sci. Math. (Szeged), 64 (1998), 627–655.
- [19] S. Hassi, M.M. Malamud, and H.S.V. de Snoo, "On Krein's extension theory of nonnegative operators", preprint, 2002.
- [20] S. Hassi and H.S.V. de Snoo, "One-dimensional graph perturbations of selfadjoint relations", Ann. Acad. Sci. Fenn. A.I. Math., 22 (1997), 123–164.
- [21] S. Hassi and H.S.V. de Snoo, "On rank one perturbations of selfadjoint operators", Integral Equations Operator Theory, 29 (1997), 288–300.
- [22] S. Hassi and H.S.V. de Snoo, "Nevanlinna functions, perturbation formulas and triplets of Hilbert spaces", Math. Nachr., 195 (1998), 115–138.
- [23] R. Horn and C. Johnson, *Matrix analysis*, Cambridge University Press, Cambridge-London-New York-Rochelle-Melbourne-Sydney, 1986.
- [24] E.J. Ionascu, "Rank-one perturbations of diagonal operators", Integral Equations Operator Theory, 39 (2001), 421–440.
- [25] I.S. Kac and M.G. Krein, "R-functions – analytic functions mapping the upper halfplane into itself", Supplement to the Russian edition of F.V. Atkinson, *Discrete and continuous boundary problems*, Mir, Moscow, 1968 (Russian) (English translation: Amer. Math. Soc. Transl. Ser. 2, 103 (1974), 1–18).
- [26] T. Kato, *Perturbation theory*, Springer-Verlag, Berlin-Heidelberg-New York, 1966.
- [27] A. Kiselev and B. Simon, "Rank one perturbations with infinitesimal coupling", J. Funct. Anal., 130 (1995), 345–356.
- [28] M.G. Krein, "The theory of selfadjoint extensions of semibounded Hermitian operators and its applications, I", Mat. Sb., 20 (1947), 431–495.
- [29] M.G. Krein and H. Langer, "Über die Q-Funktion eines  $\pi$ -hermiteschen Operators im Raum  $\Pi_\kappa$ ", Acta Sci. Math. (Szeged), 34 (1973), 191–230.
- [30] L.P. Nizhnik, "On rank one singular perturbations of selfadjoint operators", Methods of Functional Analysis and Topology, 7 (2001), no. 3, 54–66.
- [31] F. Rellich, "Störungstheorie der Spektralzerlegung I, II", Math. Ann., 113 (1937), 600–619, 677–685.

- [32] B. Simon, "Spectral analysis of rank one perturbations and applications", in J. Feldman, R. Froese and L.M. Rosen (editors), *Proceedings on Mathematical Quantum Theory II: Schrödinger operators*, CRM Proceedings and Lecture Notes, Vol. 8, Amer. Math. Soc., Providence, R.I., 1995.
- [33] I. Stakgold, *Green's functions and boundary value problems*, John Wiley and Sons, New York, 1979.
- [34] B. Szökefalvi-Nagy, "Perturbations des transformations auto-adjoints dans l'espace de Hilbert", *Comment Math. Helv.*, 19 (1947), 347–366.

DEPARTMENT OF MATHEMATICAL ANALYSIS, EAST UKRAINIAN NATIONAL UNIVERSITY, KVARTAL MOLODOZHNY 20-A, LUGANSK 91034, UKRAINE

*E-mail address:* yma@snu.edu.ua

DEPARTMENT OF MATHEMATICS AND STATISTICS, UNIVERSITY OF VAASA, P.O. Box 700, 65101 VAASA, FINLAND

*E-mail address:* sha@uwasa.fi

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF GRONINGEN, P.O. Box 800, 9700 AV GRONINGEN, NEDERLAND

*E-mail address:* desnoo@math.rug.nl

DEPARTMENT OF MATHEMATICS, NIAGARA UNIVERSITY, P.O. Box 2044, NIAGARA, NY 14109, USA

*E-mail address:* tsekanov@niagara.edu

*This page intentionally left blank*

## **Titles in This Series**

- 323 **Vadim Olshevsky, Editor**, Fast algorithms for structured matrices; theory and applications, 2003
- 322 **S. Dale Cutkosky, Dan Edidin, Zhenbo Qin, and Qi Zhang, Editors**, Vector bundles and representation theory, 2003
- 321 **Anna Kamińska, Editor**, Trends in Banach spaces and operator theory, 2003
- 320 **William Beckner, Alexander Nagel, Andreas Seeger, and Hart F. Smith, Editors**, Harmonic analysis at Mount Holyoke, 2003
- 319 **W. H. Schikhof, C. Perez-Garcia, and A. Escassut, Editors**, Ultrametric functional analysis, 2003
- 318 **David E. Radford, Fernando J. O. Souza, and David N. Yetter, Editors**, Diagrammatic morphisms and applications, 2003
- 317 **Hui-Hsiung Kuo and Ambar N. Sengupta, Editors**, Finite and infinite dimensional analysis in honor of Leonard Gross, 2003
- 316 **O. Cornea, G. Lupton, J. Oprea, and D. Tanré, Editors**, Lusternik-Schnirelmann category and related topics, 2002
- 315 **Theodore Voronov, Editor**, Quantization, Poisson brackets and beyond, 2002
- 314 **A. J. Berrick, Man Chun Leung, and Xingwang Xu, Editors**, Topology and Geometry: Commemorating SISTAG, 2002
- 313 **M. Zuhair Nashed and Otmar Scherzer, Editors**, Inverse problems, image analysis, and medical imaging, 2002
- 312 **Aaron Bertram, James A. Carlson, and Holger Kley, Editors**, Symposium in honor of C. H. Clemens, 2002
- 311 **Clifford J. Earle, William J. Harvey, and Sevin Recillas-Pishmish, Editors**, Complex manifolds and hyperbolic geometry, 2002
- 310 **Alejandro Adem, Jack Morava, and Yongbin Ruan, Editors**, Orbifolds in mathematics and physics, 2002
- 309 **Martin Guest, Reiko Miyaoka, and Yoshihiro Ohnita, Editors**, Integrable systems, topology, and physics, 2002
- 308 **Martin Guest, Reiko Miyaoka, and Yoshihiro Ohnita, Editors**, Differentiable geometry and integrable systems, 2002
- 307 **Ricardo Weder, Pavel Exner, and Benoit Grébert, Editors**, Mathematical results in quantum mechanics, 2002
- 306 **Xiaobing Feng and Tim P. Schulze, Editors**, Recent advances in numerical methods for partial differential equations and applications, 2002
- 305 **Samuel J. Lomonaco, Jr. and Howard E. Brandt, Editors**, Quantum computation and information, 2002
- 304 **Jorge Alberto Calvo, Kenneth C. Millett, and Eric J. Rawdon, Editors**, Physical knots: Knotting, linking, and folding geometric objects in  $\mathbb{R}^3$ , 2002
- 303 **William Cherry and Chung-Chun Yang, Editors**, Value distribution theory and complex dynamics, 2002
- 302 **Yi Zhang, Editor**, Logic and algebra, 2002
- 301 **Jerry Bona, Roy Choudhury, and David Kaup, Editors**, The legacy of the inverse scattering transform in applied mathematics, 2002
- 300 **Sergei Vostokov and Yuri Zarhin, Editors**, Algebraic number theory and algebraic geometry: Papers dedicated to A. N. Parshin on the occasion of his sixtieth birthday, 2002
- 299 **George Kamberov, Peter Norman, Franz Pedit, and Ulrich Pinkall, Editors**, Quaternions, spinors, and surfaces, 2002
- 298 **Robert Gilman, Alexei G. Myasnikov, and Vladimir Shpilrain, Editors**, Computational and statistical group theory, 2002

## TITLES IN THIS SERIES

- 297 **Stephen Berman, Paul Fendley, Yi-Zhi Huang, Kailash Misra, and Brian Parshall, Editors**, Recent developments in infinite-dimensional Lie algebras and conformal field theory, 2002
- 296 **Sean Cleary, Robert Gilman, Alexei G. Myasnikov, and Vladimir Shpilrain, Editors**, Combinatorial and geometric group theory, 2002
- 295 **Zhangxin Chen and Richard E. Ewing, Editors**, Fluid flow and transport in porous media: Mathematical and numerical treatment, 2002
- 294 **Robert Coquereaux, Ariel García, and Roberto Trinchero, Editors**, Quantum symmetries in theoretical physics and mathematics, 2002
- 293 **Donald M. Davis, Jack Morava, Goro Nishida, W. Stephen Wilson, and Nobuaki Yagita, Editors**, Recent progress in homotopy theory, 2002
- 292 **A. Chenciner, R. Cushman, C. Robinson, and Z. Xia, Editors**, Celestial Mechanics, 2002
- 291 **Bruce C. Berndt and Ken Ono, Editors**,  $q$ -series with applications to combinatorics, number theory, and physics, 2001
- 290 **Michel L. Lapidus and Machiel van Frankenhuyzen, Editors**, Dynamical, spectral, and arithmetic zeta functions, 2001
- 289 **Salvador Pérez-Esteva and Carlos Villegas-Blas, Editors**, Second summer school in analysis and mathematical physics: Topics in analysis: Harmonic, complex, nonlinear and quantization, 2001
- 288 **Marisa Fernández and Joseph A. Wolf, Editors**, Global differential geometry: The mathematical legacy of Alfred Gray, 2001
- 287 **Marlos A. G. Viana and Donald St. P. Richards, Editors**, Algebraic methods in statistics and probability, 2001
- 286 **Edward L. Green, Serkan Hoşten, Reinhard C. Laubenbacher, and Victoria Ann Powers, Editors**, Symbolic computation: Solving equations in algebra, geometry, and engineering, 2001
- 285 **Joshua A. Leslie and Thierry P. Robart, Editors**, The geometrical study of differential equations, 2001
- 284 **Gaston M. N'Guérékata and Asamoah Nkwanta, Editors**, Council for African American researchers in the mathematical sciences: Volume IV, 2001
- 283 **Paul A. Milewski, Leslie M. Smith, Fabian Waleffe, and Esteban G. Tabak, Editors**, Advances in wave interaction and turbulence, 2001
- 282 **Arlan Ramsay and Jean Renault, Editors**, Groupoids in analysis, geometry, and physics, 2001
- 281 **Vadim Olshevsky, Editor**, Structured matrices in mathematics, computer science, and engineering II, 2001
- 280 **Vadim Olshevsky, Editor**, Structured matrices in mathematics, computer science, and engineering I, 2001
- 279 **Alejandro Adem, Gunnar Carlsson, and Ralph Cohen, Editors**, Topology, geometry, and algebra: Interactions and new directions, 2001
- 278 **Eric Todd Quinto, Leon Ehrenpreis, Adel Faridani, Fulton Gonzalez, and Eric Grinberg, Editors**, Radon transforms and tomography, 2001
- 277 **Luca Capogna and Loredana Lanzani, Editors**, Harmonic analysis and boundary value problems, 2001
- 276 **Emma Previato, Editor**, Advances in algebraic geometry motivated by physics, 2001
- 275 **Alfred G. Noël, Earl Barnes, and Sonya A. F. Stephens, Editors**, Council for African American researchers in the mathematical sciences: Volume III, 2001
- 274 **Ken-ichi Maruyama and John W. Rutter, Editors**, Groups of homotopy self-equivalences and related topics, 2001

TITLES IN THIS SERIES

- 273 **A. V. Kelarev, R. Göbel, K. M. Rangaswamy, P. Schultz, and C. Vinsonhaler, Editors**, Abelian groups, rings and modules, 2001
- 272 **Eva Bayer-Fluckiger, David Lewis, and Andrew Ranicki, Editors**, Quadratic forms and their applications, 2000
- 271 **J. P. C. Greenlees, Robert R. Bruner, and Nicholas Kuhn, Editors**, Homotopy methods in algebraic topology, 2001
- 270 **Jan Denef, Leonard Lipschitz, Thanases Pheidas, and Jan Van Geel, Editors**, Hilbert's tenth problem: Relations with arithmetic and algebraic geometry, 2000
- 269 **Mikhail Lyubich, John W. Milnor, and Yair N. Minsky, Editors**, Laminations and foliations in dynamics, geometry and topology, 2001
- 268 **Robert Gulliver, Walter Littman, and Roberto Triggiani, Editors**, Differential geometric methods in the control of partial differential equations, 2000
- 267 **Nicolás Andruskiewitsch, Walter Ricardo Ferrer Santos, and Hans-Jürgen Schneider, Editors**, New trends in Hopf algebra theory, 2000
- 266 **Caroline Grant Melles and Ruth I. Michler, Editors**, Singularities in algebraic and analytic geometry, 2000
- 265 **Dominique Arlettaz and Kathryn Hess, Editors**, Une dégustation topologique: Homotopy theory in the Swiss Alps, 2000
- 264 **Kai Yuen Chan, Alexander A. Mikhalev, Man-Keung Siu, Jie-Tai Yu, and Efim I. Zelmanov, Editors**, Combinatorial and computational algebra, 2000
- 263 **Yan Guo, Editor**, Nonlinear wave equations, 2000
- 262 **Paul Igodt, Herbert Abels, Yves Félix, and Fritz Grunewald, Editors**, Crystallographic groups and their generalizations, 2000
- 261 **Gregory Budzban, Philip Feinsilver, and Arun Mukherjea, Editors**, Probability on algebraic structures, 2000
- 260 **Salvador Pérez-Esteva and Carlos Villegas-Blas, Editors**, First summer school in analysis and mathematical physics: Quantization, the Segal-Bargmann transform and semiclassical analysis, 2000
- 259 **D. V. Huynh, S. K. Jain, and S. R. López-Permouth, Editors**, Algebra and its applications, 2000
- 258 **Karsten Grove, Ib Henning Madsen, and Erik Kjær Pedersen, Editors**, Geometry and topology: Aarhus, 2000
- 257 **Peter A. Cholak, Steffen Lempp, Manuel Lerman, and Richard A. Shore, Editors**, Computability theory and its applications: Current trends and open problems, 2000
- 256 **Irwin Kra and Bernard Maskit, Editors**, In the tradition of Ahlfors and Bers: Proceedings of the first Ahlfors-Bers colloquium, 2000
- 255 **Jerry Bona, Katarzyna Saxton, and Ralph Saxton, Editors**, Nonlinear PDE's, dynamics and continuum physics, 2000
- 254 **Mourad E. H. Ismail and Dennis W. Stanton, Editors**,  $q$ -series from a contemporary perspective, 2000
- 253 **Charles N. Delzell and James J. Madden, Editors**, Real algebraic geometry and ordered structures, 2000
- 252 **Nathaniel Dean, Cassandra M. McZeal, and Pamela J. Williams, Editors**, African Americans in Mathematics II, 1999

For a complete list of titles in this series, visit the  
AMS Bookstore at [www.ams.org/bookstore/](http://www.ams.org/bookstore/).