A Fast Schur–Euclid-Type Algorithm for Quasiseparable Polynomials

Sirani M. Perera and Vadim Olshevsky

Abstract In this paper, a fast $\mathcal{O}(n^2)$ algorithm is presented for computing recursive triangular factorization of a Bezoutian matrix associated with quasiseparable polynomials via a displacement equation. The new algorithm applies to a fairly general class of quasiseparable polynomials that includes real orthogonal, Szegö polynomials, and several other important classes of polynomials, e.g., those defined by banded Hessenberg matrices. While the algorithm can be seen as a Schur-type for the Bezoutian matrix it can also be seen as a Euclid-type for quasiseparable polynomials via factorization of a displacement equation. The process, i.e., fast Euclid-type algorithm for quasiseparable polynomials or Schur-type algorithm for Bezoutian associated with quasiseparable polynomials, is carried out with the help of a displacement equation satisfied by Bezoutian and Confederate matrices leading to $\mathcal{O}(n^2)$ complexity.

Keywords Quasiseparable matrices · Bezoutians · Euclid algorithm · Schur algorithm · Fast algorithms · Displacement structure

1 Introduction

It is known that the Euclidean algorithm is one of the oldest algorithms which appear in the computation of the greatest common divisor (GCD). Although the original Euclidean algorithm was presented to compute the positive greatest common divisor of two given positive integers, later it was generalized to polynomials in one variable over a field, and further to polynomials in any number of variables over any unique factorization domain in which the greatest common divisor can be computed.

S.M. Perera (✉)
Embry-Riddle Aeronautical University, Daytona Beach, USA
e-mail: pereras2@erau.edu

V. Olshevsky
University of Connecticut, Storrs, USA
e-mail: olshevsky@math.uconn.edu

© Springer International Publishing AG 2017

I.S. Kotsireas and E. Martínez-Moro (eds.), *Applications of Computer Algebra*,
Springer Proceedings in Mathematics & Statistics 198,
DOI 10.1007/978-3-319-56932-1_24

1.1 GCD Computing Algorithms

The Euclidean algorithm for computing polynomial GCD evolved with the early work of Brown (see, e.g., [14, 15]) and thereafter several other authors studied: the degree of the greatest common divisor of two polynomials in connection to a companion matrix [2], stable algorithms to compute polynomial ε-GCD using the displacement structure of Sylvester and Bezout matrices [10], generalization of the Euclidean algorithm (determining the greatest common left divisor) to polynomial matrices [1], estimation of the degree of ε-GCDs at a low computational cost [41], approximate factorization of multivariate polynomials with complex coefficients containing numerical noise [35], generalized Euclidean algorithm of the Routh–Hurwitz type [19], a numeric parameter for determining two prime polynomials under small perturbations with the help of an inversion formula for Sylvester matrices [5, 6], algorithms to approximate GCD for polynomials with coefficients of floating-point numbers [39], and so on. Our intention is not to consider the Euclidean algorithm via the above methods but to see it via the Hessenberg displacement structure of a Bezoutian over a system of polynomials $\{Q\} = \{Q_k(x)\}_{k=0}^{n}$ satisfying recurrence relations having $\deg Q_k(x) = k$ and to derive a fast algorithm based on quasiseparable polynomials.

1.2 Connection to Bezoutian

Given a pair of polynomials $a(x)$ and $b(x)$ with $\deg a(x) = n$ and $\deg b(x) \le n$, the classical Bezoutian of $a(x)$ and $b(x)$ is the bilinear form given by

$$\frac{a(x)b(y) - a(y)b(x)}{x - y} = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} s_{ij} x^i y^j$$

and the Bezoutian matrix of $a(x)$ and $b(x)$ is defined by the $n \times n$ symmetric matrix $Bez(a, b) = \left[s_{ij}\right]_{i,j=0}^{n-1}$.

In the eighteenth century, the Bezoutian was invented in order to build a bridge between polynomial and linear algebra. As it was remarked in [23, 44], the Bezoutian concept in principle already evolves from Euler's work in elimination theory for polynomials. Hermite was the first who studied Bezoutians in more detail to solve root localization problems for polynomials (Routh–Hurwitz), which are important in particular for the investigation of the stability of linear systems. Note that in the early stages of work related to Bezoutians, the language of quadratic forms was more common than matrix language. After the first observation of inversion of Bezoutians as Toeplitz or Hankel by L.T. Lander in 1974, there were significant results published to show that the inverse of Toeplitz is T-Bezoutian and Hankel is H-Bezoutian (see,

e.g., [23, 26, 27, 30, 43]). These works show us great examples on the importance of matrix representations for the inverses of Hankel, Toeplitz, and more general types of structured matrices in the construction of fast algorithms for solving structured systems of equations and interpolation problems.

As this paper connects the generalized Bezoutian with confederate matrices via a Hessenberg displacement structure, we should remark heavily on Barnett's result [3] on showing the important relationship between a Bezoutian matrix and a matrix polynomial associated with the companion matrix. Apart from this, several others studied connections of Bezoutians to GCD including: computing the greatest common right divisor using Sylvester and generalized Bezoutian resultant matrices [13], matrix representations for generalized Bezoutians via generalized companion matrices [43], Bezoutians of Chebyshev polynomials of first and second kind [20], generalized Barnett factorization formula for polynomial Bezoutian matrices and reduction of Bezoutian via polynomial Vandermonde matrix [45], computation of polynomial GCD and coefficients of the polynomials generated in the Euclidean scheme via Bezoutian [11], computation of the GCD of two polynomials using Bernstein–Bezoutian matrix [12], and so on.

1.3 Connection to Displacement Structure

This paper describes a fast Euclid-type algorithm for quasiseparable polynomials via a fast Schur-type algorithm for a Bezoutian matrix preserving a Hessenberg displacement structure. The structured matrices like Toeplitz, Hankel, Toeplitz plus Hankel, Vandermonde, Cauchy, etc. belong to a more general family of matrices with low rank displacement structure and that can be used to design fast algorithms.

Definition 1 A linear displacement operator $\Theta_{\Omega,M,F,N}(.) : C^{n\times n} \to C^{n\times n}$ is a function which transforms each matrix $R \in C^{n\times n}$ to the matrix given by the displacement equation

$$\Theta_{\Omega,M,F,N}(R) = \Omega\, RM - FRN = GB \tag{1}$$

where $\Omega, M, F, N \in C^{n\times n}$ are given matrices and $G \in C^{n\times\alpha}$, $B \in C^{\alpha\times n}$. The pair $\{G, B\}$ on last right in (1) is called a minimal generator of R and

$$rank\ \left\{\Theta_{\Omega,M,F,N}(R)\right\} = \alpha. \tag{2}$$

Example 1 Toeplitz matrix $T = [t_{i-j}]_{1\le i,j\le n}$ satisfies the displacement equation

$$T - Z \cdot T \cdot Z^T = \begin{bmatrix} t_0 & t_{-1} & \cdots & t_{-n+1} \\ t_1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ t_{n-1} & 0 & \cdots & 0 \end{bmatrix}$$
$$= \begin{bmatrix} \frac{t_0}{2} & 1 \\ t_1 & 0 \\ \vdots & \vdots \\ t_{n-1} & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & \cdots 0 \\ \frac{t_0}{2} & t_{-1} & \cdots t_{-n+1} \end{bmatrix}$$

where Z is a lower shift matrix. Thus, rank $\left\{\Theta_{I,I,Z,Z^T}(T)\right\} = 2$.

Example 2 Hankel matrix $H = [h_{i+j-2}]_{1 \le i,j \le n}$ satisfies the displacement equation

$$Z \cdot H - H \cdot Z^T = \begin{bmatrix} 0 & -h_0 & -h_1 & \cdots & -h_{n-2} \\ h_0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{n-2} & 0 & 0 & \cdots & 0 \end{bmatrix}$$
$$= \begin{bmatrix} 1 & 0 \\ 0 & h_0 \\ \vdots & \vdots \\ 0 & h_{n-2} \end{bmatrix} \begin{bmatrix} 0 & -h_0 & \cdots & -h_{n-2} \\ 1 & 0 & \cdots & 0 \end{bmatrix}$$

where Z is a lower shift matrix. Thus, rank $\left\{\Theta_{Z,I,I,Z^T}(H)\right\} = 2$.

A fast algorithm for the structured matrices which preserve displacement structure first appeared in Morf's Thesis [38]. Thus the crucial shift-low-rank updating property was recognized by the author as the proper generalization of the Toeplitz and Hankel structured matrices. The algorithm was called Fast Cholesky decomposition. In June 1971, computer programs were successfully completed by the author. In the same Thesis he announced a divide-and-conquer algorithm but it was not shown how to design a super fast algorithm. Such an algorithm was obtained in Brent-Gustafson-Yun in 1979. Moreover the paper of Kailath et al. [31] proved crucial results demonstrating that the Schur complement inherits the displacement rank. This idea was the opening of a new chapter, as it filled in the missing link of proving a super fast complexity in Morf's Thesis. Delosme in [17] obtained formulas for generator updates for the Toeplitz case and claimed that those coincided with the classical Schur algorithm. Furthermore one can find (e.g., in [18, 32–34]) algorithms that connect structured matrices with displacement equations to derive fast algorithms. Many polynomial computations can be reduced to structured matrix computations. In this way, the matrix interpretation of many classical polynomial algorithms for determining GCD can be expressed.

The displacement equations of the structured matrices are used to design fast Schur-type algorithms having complexity $\mathcal{O}(n^2)$. The existence of fast Schur-type algorithms for Toeplitz and Toeplitz-like matrices having low displacement rank was shown in [31, 38]. It was shown in [25] that the low displacement rank Vandermonde-like and Cauchy-like matrices can be used to derive fast Schur-type algorithms. We should also recall the fast $\mathcal{O}(n^2)$ algorithm for Cauchy-like displacement structured matrices via Gaussian elimination with partial pivoting and fast algorithms for Toeplitz-like, Toeplitz-plus-Hankel-like, Vandermonde-like matrices via transferring those matrices to Cauchy-like matrices in [21]. Moreover in [40] a fast Schur-type algorithm with stability criteria was presented for the factorization of Hankel-like matrices. Finally, the crucial result of the Schur-type algorithm was presented by Heinig and Olshevsky [24] for the matrices with Hessenberg displacement structure.

While the displacement structure is considered we should recall some results on Schur-type algorithms in connection to the Bezoutian. At this point, we should mention the results on: computing Schur type and hybrid (Schular type and Levinson type) algorithms to solve the system of equations involving Toeplitz-plus-Hankel matrices [29], computing a Schur-type algorithm for LDU-decomposing the strongly regular Toeplitz-plus-Hankel matrix [46], solving a system of equations by split algorithms for skewsymmetric Toeplitz matrices, centrosymmetric Toeplitz-plus-Hankel matrices, and general Toeplitz-plus-Hankel matrices [28], and more importantly, Olshevsky's claim on Schur-type algorithms in connection to Euclid-type algorithms via the Bezoutian in the 10th ILAS Conference in 2002 and 16th International Symposium on Mathematical Theory of Networks and Systems in 2004.

1.4 Main Results

In [24], a Schur-type algorithm was presented to compute a recursive triangular factorization $R = LDU$ for a strongly nonsingular $n \times n$ matrix R satisfying the displacement equation:

$$RY - VR = GH^T$$

with upper and lower Hessenberg matrices Y and V, respectively, and $n \times \alpha$ matrices G and H where α is small compared to n. The Schur–Hessenberg algorithm in [24] will have complexity $\mathcal{O}(n^3)$ in general for dense and unstructured Hessenberg matrices Y and V. However, one can explore the structures of Y and V to derive a $\mathcal{O}(n^2)$ algorithm. Thus in this paper, we explore the structures of Y and V to derive a fast hybrid of Schur-type and Euclid-type algorithms in connection with Bezoutian and confederate matrices over the system of quasiseparable polynomials.

We observe a displacement equation of a Bezoutian matrix associated with the reverse polynomials in connection to a companion matrix over the system of monomial basis (this displacement equation (14) is a variant of the Lancaster–Tismenetsky equation in [36]). We then use this to derive and generalize a displacement equation for a generalized Bezoutian matrix with confederate matrix respect to the system of

polynomials $\{Q\} = \{Q_k(x)\}_{k=0}^{n}$ satisfying recurrence relations having deg $Q_k(x) = k$. Then, the displacement equation for a generalized Bezoutian with confederate matrix, and characteristics of the Schur complement of the generalized Bezoutian and generator updates for confederate and generalized Bezoutian matrices are used to derive the Schur–Euclid–Hessenberg algorithm. Finally, to derive a fast $\mathcal{O}(n^2)$ complexity Schur–Euclid–Hessenberg algorithm we take quasiseparable polynomials as the main tool.

Definition 2 A matrix $A = \left[a_{ij}\right]$ is called $(H, 1)$-quasiseparable (i.e., Hessenberg-1-quasiseparable) if **(i)** it is strongly upper Hessenberg ($a_{i+1,i} \neq 0$ for $i = 1, 2, \cdots, n-1$ and $a_{i,j} = 0$ for $i > j+1$), and **(ii)** max (rank A_{12}) = 1 where the maximum is taken over all symmetric partitions of the form

$$A = \left[\begin{array}{c|c} \star & A_{12} \\ \hline \star & \star \end{array}\right]$$

• Let $A = [a_{ij}]$ be a $(H, 1)$-quasiseparable matrix. For $\alpha_i = \frac{1}{a_{i+1,i}}$, then the system of polynomials related to A via

$$r_k(x) = \alpha_1\alpha_2 \cdots \alpha_k det(xI - A)_{(k\times k)}$$

is called a system of $(H, 1)$-quasiseparable polynomials. In the classification paper [9], the characterization of orthogonal polynomials (orthogonal with respect to a weighted inner product (definite or indefinite) on the real line) and Szegö polynomials (orthogonal on the unit circle with respect to a weighted inner product) via tridiagonal and unitary Hessenberg matrices, respectively, are observed to belong to a wider class of $(H, 1)$-quasiseparable polynomials and matrices, respectively. Hence once a fast $\mathcal{O}(n^2)$ Schur–Euclid-type algorithm for quasiseparable polynomials is established, we analyze the complexity of Schur–Euclid-type algorithms for orthogonal and Szegö polynomials.

1.5 Structure of the Paper

The structure of the paper is as follows. In the next Sect. 2, we state polynomial division in a matrix form, arithmetic complexity, and see the connection to the Euclidean algorithm in matrix forms. In Sect. 3, we express displacement equations and characterizations of the Bezoutian matrix in connection to the Schur complement and reverse polynomials. Then in Sect. 4, we generalize the displacement equation in the former section via generalized Bezoutians and confederate matrices over the system of polynomials $\{Q\} = \{Q_k(x)\}_{k=0}^{n}$ satisfying recurrence relations having deg $Q_k(x) = k$. At the end of the section, we present a hybrid of Euclid-type algorithm and Schur-type algorithm using the Hessenberg displacement structure of the Bezoutian and call it the Schur–Euclid–Hessenberg algorithm. Finally in Sect. 5, we establish

a fast $\mathcal{O}(n^2)$ Schur–Euclid–Hessenberg algorithm for quasiseparable polynomials while addressing the complexity of the algorithm for its subclasses: orthogonal and Szegö polynomials.

2 One Way to Express Polynomial Division in Matrix Form

In this section, we state polynomial division in a matrix form. In the meantime, we discuss the arithmetic cost of computing the polynomial division and the matrix form of the Euclidean algorithm in connection to polynomials. Similar to this approach one can see the results in [42] to compute polynomial division efficiently. We first give the Euclidean algorithm for computing the GCD of polynomials.

Let $a(x)$ and $b(x)$ be given with $\deg a(x) \geq \deg b(x)$; then the Euclidean algorithm applies to $a(x)$ and $b(x)$ and generates a sequence of polynomials $r^{(i)}(x), q^{(i-1)}(x)$, such that

$$\begin{array}{c} r^{(0)}(x) = a(x), \qquad r^{(1)}(x) = b(x) \\ r^{(i-2)}(x) = q^{(i-1)}(x) r^{(i-1)}(x) + r^{(i)}(x), \qquad i = 2, 3, \cdots, t+1 \end{array} \tag{3}$$

where $r^{(i)}(x)$ is the remainder of the division of $r^{(i-2)}(x)$ by $r^{(i-1)}(x)$. The algorithm stops when a remainder $r^{(t+1)}(x) = 0$ is found; then $r^{(t)}(x)$ is the desired GCD of $a(x)$ and $b(x)$. Note that since the $\deg r^{(0)}(x) > \deg r^{(1)}(x) > \cdots > \deg r^{(t+1)}(x)$, the algorithm must terminate in a finite number of steps. If $r^{(t)}(x)$ is a constant then $r^{(0)}(x)$ and $r^{(1)}(x)$ are said to be relatively prime.

The following results show polynomial division (one step of a variant of Euclidean algorithm) in matrix form. Here we have considered $-c(x)$ as the remainder of the polynomial division of $a(x)$ by $b(x)$ just to be compatible with future discussions.

Lemma 1 *Let* $a(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_0$ *and* $b(x) = b_{n-k} x^{n-k} + b_{n-k-1} x^{n-k-1} + \cdots + b_0$ *where* $k \geq 1$. *Then the polynomial division of* $a(x)$ *by* $b(x)$ *can be seen via:*

$$-\begin{bmatrix} 0 \\ c_{n-k-1} \\ c_{n-k-2} \\ \vdots \\ c_0 \end{bmatrix} + q_0 \begin{bmatrix} b_{n-k} \\ b_{n-k-1} \\ b_{n-k-2} \\ \vdots \\ b_0 \end{bmatrix} = \begin{bmatrix} a_{n-k} \\ a_{n-k-1} \\ a_{n-k-2} \\ \vdots \\ a_0 \end{bmatrix} - \widehat{B_k} B_k^{-1} \begin{bmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_{n-k+1} \end{bmatrix} \tag{4}$$

where $-c(x) = -c_{n-k-1} x^{n-k-1} - c_{n-k-2} x^{n-k-2} - \cdots - c_0$ *is the remainder,* q_0 *is the constant term of the quotient of polynomial division,*

$$B_k := toeplitz\left([b_{n-k} : b_{n-2k+1}], [b_{n-k},\ zeros(1, k-1)]\right),$$

and

$$\widehat{B_k} = \begin{bmatrix} b_{n-2k} & b_{n-2k+1} & b_{n-2k+2} & & \cdots & b_{n-k-1} \\ b_{n-2k-1} & b_{n-2k} & b_{n-2k+1} & & \cdots & \vdots \\ b_{n-2k-2} & b_{n-2k-1} & b_{n-2k} & & & \\ \vdots & \vdots & \vdots & & & \vdots \\ b_0 & b_1 & \vdots & & & \\ 0 & b_0 & b_1 & & & \\ 0 & 0 & \ddots & \ddots & & \vdots \\ & & \ddots & \ddots & & \\ \vdots & & & 0 & b_0 & b_1 \\ & & & & 0 & b_0 \\ 0 & & \cdots & & & 0 \end{bmatrix}.$$

Proof If $q(x) = q_k x^k + q_{k-1}x^{k-1} + \cdots + q_0$ and $-c(x)$ are the quotient and remainder of the polynomial division of $a(x)$ by $b(x)$ then we can say

$$\begin{aligned} a_n x^n + a_{n-1}x^{n-1} + \cdots + a_0 = (q_k x^k + q_{k-1}x^{k-1} + \cdots + q_0) \\ \cdot(b_{n-k}x^{n-k} + b_{n-k-1}x^{n-k-1} + \cdots + b_0) \\ -(c_{n-k-1}x^{n-k-1} + c_{n-k-2}x^{n-k-2} + \cdots + c_0) \end{aligned} \quad (5)$$

By equating the coefficients of x^n to x^{n-k+1} in (5);

$$\begin{bmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_{n-k+2} \\ a_{n-k+1} \end{bmatrix} = \begin{bmatrix} b_{n-k} & 0 & \cdots & \cdots & 0 \\ b_{n-k-1} & b_{n-k} & \ddots & & \vdots \\ b_{n-k-2} & b_{n-k-1} & b_{n-k} & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ b_{n-2k+1} & b_{n-2k+2} & \ldots & b_{n-k-1} & b_{n-k} \end{bmatrix} \cdot \begin{bmatrix} q_k \\ q_{k-1} \\ \vdots \\ q_2 \\ q_1 \end{bmatrix} \quad (6)$$

Note that the first matrix in the RHS of (6) is a lower Toeplitz matrix (say B_k where $B_k := toeplitz\,([b_{n-k} : b_{n-2k+1}], [b_{n-k},\ zeros(1, k-1)])$ so q_k's (except q_0) can be recovered from:

$$\begin{bmatrix} q_k \\ q_{k-1} \\ \vdots \\ q_2 \\ q_1 \end{bmatrix} = [B_k]^{-1} \cdot \begin{bmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_{n-k+2} \\ a_{n-k-1} \end{bmatrix} \quad (7)$$

Equating coefficients of x^{n-k} to the constant term in (5);

$$-\begin{bmatrix} 0 \\ c_{n-k-1} \\ c_{n-k-2} \\ \vdots \\ c_0 \end{bmatrix} = \begin{bmatrix} a_{n-k} \\ a_{n-k-1} \\ \vdots \\ a_0 \end{bmatrix} - \begin{bmatrix} b_{n-2k} & b_{n-2k+1} & b_{n-2k+2} & \cdots & & b_{n-k} \\ b_{n-2k-1} & b_{n-2k} & b_{n-2k+1} & \cdots & & b_{n-k-1} \\ \vdots & \vdots & \vdots & & & \vdots \\ b_1 & \vdots & \vdots & & & \vdots \\ b_0 & b_1 & \vdots & & & \\ 0 & b_0 & b_1 & & & \vdots \\ 0 & 0 & b_0 & b_1 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \ddots & b_1 \\ 0 & \cdots & \cdots & 0 & 0 & b_0 \end{bmatrix} \cdot \begin{bmatrix} q_k \\ q_{k-1} \\ q_{k-2} \\ \vdots \\ q_0 \end{bmatrix}$$

By rearranging the above system we get

$$-\begin{bmatrix} 0 \\ c_{n-k-1} \\ c_{n-k-2} \\ \vdots \\ c_0 \end{bmatrix} + q_0 \begin{bmatrix} b_{n-k} \\ b_{n-k-1} \\ \vdots \\ b_0 \end{bmatrix}$$

$$= \begin{bmatrix} a_{n-k} \\ a_{n-k-1} \\ \vdots \\ a_0 \end{bmatrix} - \begin{bmatrix} b_{n-2k} & b_{n-2k+1} & b_{n-2k+2} & \cdots & b_{n-k-1} \\ b_{n-2k-1} & b_{n-2k} & b_{n-2k+1} & \cdots & b_{n-k-2} \\ \vdots & \vdots & \vdots & & \vdots \\ b_1 & \vdots & \vdots & & \vdots \\ b_0 & b_1 & \vdots & & \\ 0 & b_0 & b_1 & & \vdots \\ 0 & 0 & b_0 & b_1 & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \ddots & b_1 \\ 0 & \cdots & \cdots & 0 & 0 & b_0 \\ 0 & \cdots & \cdots & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} q_k \\ q_{k-1} \\ q_{k-2} \\ \vdots \\ q_1. \end{bmatrix}$$

However we can now use (7) to rewrite the right side of the above equation and which yields the result (4).

Corollary 1 *Let $a(x)$ and $b(x)$ be two polynomials such that deg $a(x) =$ deg $b(x) + 1$ with deg $a(x) = n$. Then the polynomial division of $a(x)$ by $b(x)$ can be seen via:*

$$-\begin{bmatrix} 0 \\ c_{n-2} \\ \vdots \\ c_1 \\ c_0 \end{bmatrix} + q_0 \begin{bmatrix} b_{n-1} \\ b_{n-2} \\ \vdots \\ b_0 \end{bmatrix} = \begin{bmatrix} a_{n-1} \\ a_{n-2} \\ \vdots \\ a_0 \end{bmatrix} - q_1 Z^T \begin{bmatrix} b_{n-1} \\ b_{n-2} \\ \vdots \\ b_1 \\ b_0 \end{bmatrix}$$

where $q_1 = a_n\, b_{n-1}^{-1}$, *and* Z *is the lower shift matrix.*

The following gives the Toeplitz matrix-based calculation of the quotient of the polynomial division and its arithmetic cost.

Corollary 2 *If* $a(x)$ *and* $b(x)$ *are two polynomials such that deg* $a(x) = n$ *and deg* $b(x) = n - k$ *where* $k \geq 1$, *then the arithmetic cost of computing the quotient of the polynomial division is* $\mathcal{O}(n \log n)$ *operations.*

Proof Let $q(x)$ be the quotient of the polynomial division of $a(x)$ by $b(x)$ stated via (5). Then the coefficients of quotient, q_k, can be recovered via

$$\begin{aligned} \begin{bmatrix} q_k \\ q_{k-1} \\ \vdots \\ q_1 \\ q_0 \end{bmatrix} &= \begin{bmatrix} b_{n-k} & 0 & \cdots & \cdots & 0 \\ b_{n-k-1} & b_{n-k} & \ddots & & \vdots \\ b_{n-k-2} & b_{n-k-1} & b_{n-k} & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ b_{n-2k} & b_{n-2k+1} & \ldots & b_{n-k-1} & b_{n-k} \end{bmatrix}^{-1} \cdot \begin{bmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_{n-k+1} \\ a_{n-k} \end{bmatrix} \\ &= B_{k+1}^{-1} \cdot \begin{bmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_{n-k+1} \\ a_{n-k} \end{bmatrix} \end{aligned} \tag{8}$$

Note that $B_{k+1} = b_{n-k} I + b_{n-k-1} Z + \cdots + b_{n-2k} Z^k = \bar{b}(Z)$ where Z is the lower shift matrix and $Z^{k+1} = 0$. Thus $B_{k+1}^{-1} = \bar{b}(Z)^{-1} mod\ \ Z^{k+1}$. Note that B_{k+1}^{-1} is also a lower triangular Toeplitz matrix which is defined by its first column. Now by following [42], one can apply a divide-and-conquer technique for the block form of the B_{k+1}^{-1} to calculate the first column of B_{k+1}^{-1}. This yields the cost of computing B_{k+1}^{-1} and also a Toeplitz matrix times a vector is of order $\mathcal{O}(n\ \log n)$.

The following shows the cost of computing sequences of remainder polynomials via a variant of the Euclidean algorithm which corresponds to polynomial division in matrix form.

Corollary 3 *If the sequence of remainders of the polynomial division is computed via Lemma 1, then the cost of computing one division is* $\mathcal{O}(n \log^2 n)$ *and* $\mathcal{O}(n\, t\ \log^2 n)$ *for generating the full sequence where n is the degree of the divisor and t is the number of steps.*

Proof As stated in Corollary 2, the cost of computing the quotient of the polynomial division is $\mathcal{O}(n \log n)$. Thus computing $B_k^{-1}\bar{a}$ where $\bar{a} = \begin{bmatrix} a_n & a_{n-1} & \cdots & a_{n-k+1} \end{bmatrix}^T$ costs $\mathcal{O}(n \log n)$ operations. Now the multiplication of $B_k^{-1}\bar{a}$ by a tall sparse matrix $\widehat{B}_k$ together with vector subtraction yields $\mathcal{O}(n \log^2 n)$ operations for one division or one step in calculating the remainder. Thus to generate t steps or for a full sequence it costs $\mathcal{O}(n\,t \log^2 n)$ operations.

The next section shows the displacement equations of Bezoutian and Schur complement in connection to reverse polynomials.

3 Displacement Structures and Characterizations of Bezoutian

This section describes two types of displacement equations of the Bezoutian while introducing characterization of the Bezoutian via Gaussian elimination and Schur complement. These displacement equations of Bezoutians are elaborated in connection to a lower shift matrix and a companion matrix but associated with the reverse polynomials.

Definition 3 Let $a(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_0$. Then, the reverse polynomial of a(x) is defined as $a^\sharp(x) = x^n a(x^{-1}) = a_0 x^n + a_1 x^{n-1} + \ldots + a_{n-1}x + a_n$.

We define the Bezoutian associated with the reverse polynomials as follows.

Definition 4 Let $P = \{1, x, x^2, ..., x^n\}$ be a monomial basis and let $a(x)$ and $b(x)$ be polynomials of degree not greater than n. Then a matrix $S^\sharp = [s_{ij}]$ is the Bezoutian associated with the reverse polynomials $a^\sharp(x)$ and $b^\sharp(x)$, say $S^\sharp = Bez_P(a^\sharp, b^\sharp)$, if

$$S(a^\sharp, b^\sharp) = \frac{a^\sharp(x)\cdot b^\sharp(y) - b^\sharp(x)\cdot a^\sharp(y)}{x-y} = \sum_{i,j=0}^{n-1} s_{ij}x^i y^j$$

$$= \begin{bmatrix} 1 & x & x^2 & \cdots & x^{n-1} \end{bmatrix} S^\sharp \begin{bmatrix} 1 \\ y \\ y^2 \\ \vdots \\ y^{n-1} \end{bmatrix}. \quad (9)$$

3.1 Displacement Structures of Bezoutian

Here we obtain two types of displacement equations of the Bezoutian associated with the reverse polynomials. Once we establish the displacement structures, we state

connections of the displacement equations to the Gohberg, Kailath, and Olshevsky algorithm (GKO algorithm) [21] and the Heinig and Olshevsky algorithm (HO algorithm) [24].

Below we give the GKO algorithm for matrix R_1 satisfying the Sylvester displacement equation.

Lemma 2 *Let matrix* $R_1 = \begin{bmatrix} r_1 & R_{12} \\ R_{21} & R_{22} \end{bmatrix}$ *satisfy:*

$$\Delta_{F_1,A_1}(R_1) = \begin{bmatrix} f_1 & 0 \\ * & F_2 \end{bmatrix} \cdot R_1 - R_1 \cdot \begin{bmatrix} a_1 & * \\ 0 & A_2 \end{bmatrix} = G^{(1)}B^{(1)} = \begin{bmatrix} g_1 \\ G_1 \end{bmatrix} \begin{bmatrix} b_1 & B_1 \end{bmatrix}$$

If r_1 *(i.e., (1,1) entry of* R_1*)* $\neq 0$*, then the Schur complement* $R_2 = R_{22}^{(1)} - R_{21}\frac{1}{r_1}R_{12}$ *satisfies the Sylvester type displacement equation*

$$F_2 \cdot R_2 - R_2 \cdot A_2 = G^{(2)}B^{(2)}$$

with

$$G_2^{(2)} = G_1 - R_{21}\frac{1}{r_1}g_1, \qquad B_2^{(2)} = B_1 - b_1\frac{1}{r_1}R_{12}$$

where g_1 *and* b_1 *are the first row of* $G^{(1)}$ *and the first column of* $B^{(1)}$ *respectively.*

The Lemma 2 shows that if R_1 satisfies a Sylvester type displacement equation then so does its Schur complement. Thus the displacement equation of the Schur complement of the matrix will also yield the factorization. One can recover the first row and column of R_1, and R_2, by using generator updates. Proceeding recursively one finally obtains the LU factorization of R_1. Moreover authors in [21] note that one can obtain a fast Gaussian elimination algorithm with partial pivoting for Cauchy-like, Vandermonde-like, and Chebyshev-like displacement structures.

Lemma 3 *Let the matrix* R_1 *satisfy:*

$$\Delta_{F_1,A_1}(R_1) = F_1 \cdot R_1 - R_1 \cdot A_1 = G^{(1)}B^{(1)}$$

and let the matrices be partitioned as

$$R_1 = \begin{bmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{bmatrix}, \quad F_1 = \begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix}, \quad A_1 = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix},$$

$$G^{(1)} = \begin{bmatrix} G_1 \\ G_2 \end{bmatrix}, \quad B^{(1)} = \begin{bmatrix} B_1 & B_2 \end{bmatrix}.$$

If R_{11} *is nonsingular, then the Schur complement of* R_1*, i.e.,* $R_2 = R_{22} - R_{21}R_{11}^{-1}R_{12}$ *satisfies*

$$F_2 \cdot R_2 - R_2 \cdot A_2 = G^{(2)}B^{(2)}$$

with

$$G^{(2)} = G_2 - R_{21}R_{11}^{-1}G_1,\ \ B^{(2)} = B_2 - B_1R_{11}^{-1}R_{12}$$
$$A_2 = A_{22} - A_{21}R_{11}^{-1}R_{12},\ \ F_2 = F_{22} - R_{21}R_{11}^{-1}F_{12}.$$

From the Lemma 3, one can observe that a Schur-type algorithm can be designed for nontriangular matrices $\{F_1, A_1\}$. Authors in [24] specialize this crucial result by deriving a Gaussian elimination algorithm for matrices with Hessenberg displacement structure. Although the algorithm has complexity $\mathcal{O}(n^3)$, in general one can explore the structures of F_1 and A_1 to derive fast algorithms.

We first observe a Sylvester type displacement equation for the Bezoutian associated with reverse polynomials in connection to the lower shift matrix and see it as a GKO-type displacement equation, but in our case it is for the Bezoutian.

Lemma 4 *Let $S^\sharp = Bez_P(a^\sharp, b^\sharp)$ be the Bezoutian associated with the reverse polynomials where $a(x) = a_nx^n + a_{n-1}x^{n-1} + \cdots + a_0$ and $b(x) = b_nx^n + b_{n-1}x^{n-1} + \cdots + b_0$ then $S^\sharp$ satisfies the displacement equation:*

$$ZS^\sharp - S^\sharp Z^T = GJG^T \tag{10}$$

where $G = \begin{bmatrix} a_n & b_n \\ a_{n-1} & b_{n-1} \\ \vdots & \vdots \\ a_1 & b_1 \end{bmatrix}$, $J = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$ *and Z is the lower shift matrix.*

Proof By following the Definition 4 of the Bezoutian associated with the reverse polynomials, we get:

$$(x-y)S(a^\sharp, b^\sharp) = a^\sharp(x)\cdot b^\sharp(y) - b^\sharp(x)\cdot a^\sharp(y). \tag{11}$$

Observing the RHS:

$$a^\sharp(x)b^\sharp(y) - b^\sharp(x)a^\sharp(y) = \begin{bmatrix} 1 & x & \cdots & x^n \end{bmatrix} \begin{bmatrix} a_n & b_n \\ a_{n-1} & b_{n-1} \\ \vdots & \vdots \\ a_0 & b_0 \end{bmatrix} \begin{bmatrix} b_n & b_{n-1} & \cdots & b_0 \\ -a_n & -a_{n-1} & \cdots & -a_0 \end{bmatrix} \begin{bmatrix} 1 \\ y \\ \vdots \\ y^n \end{bmatrix}$$

$$= \begin{bmatrix} 1 & x & x^2 & \cdots & x^n \end{bmatrix} \widetilde{G}J\widetilde{G}^T \begin{bmatrix} 1 \\ y \\ y^2 \\ \vdots \\ y^n \end{bmatrix} \tag{12}$$

where $\widetilde{G} = \begin{bmatrix} a_n & b_n \\ a_{n-1} & b_{n-1} \\ \vdots & \vdots \\ a_0 & b_0 \end{bmatrix}$ and $J = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$. Let's pad $S^\sharp$ with zeros such that $\begin{bmatrix} S^\sharp & 0 \\ 0 & 0 \end{bmatrix} = \widetilde{S}$. We can always use $\widetilde{S}$ instead of $S^\sharp$ because

$$\begin{bmatrix} 1 & x & x^2 & \cdots & x^{n-1} \end{bmatrix} S^\sharp \begin{bmatrix} 1 \\ y \\ y^2 \\ \vdots \\ y^{n-1} \end{bmatrix} = \begin{bmatrix} 1 & x & x^2 & \cdots & x^n \end{bmatrix} \widetilde{S} \begin{bmatrix} 1 \\ y \\ y^2 \\ \vdots \\ y^n \end{bmatrix}$$

By following (9) together with $\widetilde{S}$ we get:

$$(x-y)S(a^\sharp, b^\sharp) = \begin{bmatrix} x & x^2 & \cdots & x^{n+1} \end{bmatrix} \widetilde{S} \begin{bmatrix} 1 \\ y \\ \vdots \\ y^n \end{bmatrix} - \begin{bmatrix} 1 & x & \cdots & x^n \end{bmatrix} \widetilde{S} \begin{bmatrix} y \\ y^2 \\ \vdots \\ y^{n+1} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & x & x^2 & \cdots & x^n \end{bmatrix} (Z\widetilde{S} - \widetilde{S}Z^T) \begin{bmatrix} 1 \\ y \\ y^2 \\ \vdots \\ y^n \end{bmatrix}$$

Following the immediate result with (11) and (12)

$$Z\widetilde{S} - \widetilde{S}Z^T = \widetilde{G}J\widetilde{G}^T. \tag{13}$$

In the relation (13) one can peel off the last row of the generator $\widetilde{G}$, and peel off the last row and column of $\widetilde{S}$ resulting in $S^\sharp$, hence the result.

The following result is an immediate consequence of the Bezoutian satisfying the displacement equation (10) in connection to the displacement rank.

Corollary 4 *If the Bezoutian $S^\sharp$ satisfies the displacement equation* (10)*, then rank* $\left\{\Theta_{Z,I,I,Z^T}(S^\sharp)\right\} = 2$.

By following the GKO algorithm [21], one can claim that the displacement equation (10) is of GKO-type but for the Bezoutian having low displacement rank.

Next, we see the second displacement equation of the Bezoutian associated with the reverse polynomials satisfying Hessenberg displacement structure.

Lemma 5 *A matrix $S^\sharp$ is a Bezoutian for reverse polynomials $a^\sharp(x)$ and $b^\sharp(x)$ if and only if it satisfies the equation*

$$C_a^T S^\sharp - S^\sharp C_a = 0. \tag{14}$$

for a matrix C_a of the form

$$C_a = \begin{bmatrix} -\frac{a_{n-1}}{a_n} & -\frac{a_{n-2}}{a_n} & -\frac{a_{n-3}}{a_n} & \cdots & -\frac{a_0}{a_n} \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix} \tag{15}$$

where $a(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_0$ and $b(x) = b_n x^n + b_{n-1}x^{n-1} + \cdots + b_0$.

Proof Let $S^\sharp$ be the Bezoutian associated with the reverse polynomials $a^\sharp(x)$ and $b^\sharp(x)$ and C_a have the above structure (entries in the first row are extracted from the coefficients of $a(x)$) then it can easily be seen by matrix multiplication that

$$C_a^T S^\sharp - S^\sharp C_a = 0. \tag{16}$$

Notice that this is a variant of the Lancaster–Tismenetsky equation in [36].

Now let $C_a^T S^\sharp - S^\sharp C_a = 0$ where $S^\sharp = [s_{ij}]$ and C_a is the matrix of the given form so one can recover the columns of $S^\sharp$ as follows.

the second column of $S^\sharp$ by:

$$C_a^T s_{i,1} + \frac{a_{n-1}}{a_n} s_{i,1} - s_{i,2} = 0 \Rightarrow s_{i,2} = C_a^T s_{i,1} + \frac{a_{n-1}}{a_n} s_{i,1}$$

$$s_{i,2} = s_{1,1}\,\mathbf{a} + \left(Z^T + \frac{a_{n-1}}{a_n}\right) s_{i,1}$$

the third column of $S^\sharp$ by:

$$C_a^T s_{i,2} + \frac{a_{n-2}}{a_n} s_{i,1} - s_{i,3} = 0 \Rightarrow s_{i,3} = C_a^T s_{i,2} + \frac{a_{n-2}}{a_n} s_{i,1}$$

$$\begin{aligned} s_{i,3} &= s_{1,2}\,\mathbf{a} + Z^T s_{i,2} + \frac{a_{n-2}}{a_n} s_{i,1} \\ &= \left(Z^T s_{1,1} + s_{1,2} I_n\right)\mathbf{a} + \left(\left(Z^T\right)^2 + \frac{a_{n-1}}{a_n} Z^T + \frac{a_{n-2}}{a_n}\right) s_{i,1} \end{aligned}$$

proceeding recursively the kth column of $S^\sharp$ by:

$$s_{i,k} = \left(\left(Z^T\right)^{k-2} s_{1,1} + \left(Z^T\right)^{k-3} s_{1,2} + \cdots + s_{1,k-1} I_n\right) \mathbf{a} + \left(\left(Z^T\right)^{k-1} + \frac{a_{n-1}}{a_n}\left(Z^T\right)^{k-2} + \frac{a_{n-2}}{a_n}\left(Z^T\right)^{k-3} + \cdots + \frac{a_{n-k+1}}{a_n}\right) s_{i,1}$$

where $\mathbf{a} = \left[-\frac{a_{n-1}}{a_n} \; -\frac{a_{n-2}}{a_n} \cdots -\frac{a_0}{a_n}\right]^T$ and I_n is the identity matrix. Hence once all columns are recovered it should be clear that since $S^\sharp$ satisfies (16) there is no other matrix satisfying (16) and having the same first column s_{i1}. Thus $S^\sharp = Bez_P(a^\sharp, b^\sharp)$.

The displacement equation (16) is of HO-type but for the Bezoutian associated with reverse polynomials satisfying Hessenberg displacement structure in connection to the companion matrix C_a.

3.2 Characterization of Bezoutian

In this section, we perform Gaussian elimination on a Bezoutian satisfying displacement structure (10) and then elaborate on the relationship between a Bezoutian with its Schur complement.

Before we see the Schur complement of a Bezoutian as a Bezoutian, let us provide a supportive result to see how the displacement equation (13) helps us to address the rank of a Bezoutian as it is padded with zeros.

Lemma 6 *A matrix $S^\sharp = Bez_P(a^\sharp, b^\sharp) \in R^{n,n}$ is a Bezoutian if and only if $\widetilde{S} = \begin{bmatrix} S^\sharp & 0 \\ 0 & 0 \end{bmatrix}$ has displacement rank* 2.

Proof Lemma 4 suggests that if $S^\sharp$ is a Bezoutian then $\widetilde{S}$ has displacement rank 2. Conversely, if $\widetilde{S}$ has displacement rank 2, then $Z\widetilde{S} - \widetilde{S}Z^T = \widetilde{G}J\widetilde{G}^T$ for some $\widetilde{G} \in R^{n+1,2}$. Now, assume we have two polynomials and we wish to compute the Bezoutian associated with them. Lemma 4 suggests that we can do this by writing the polynomials as the columns of the generator and recovering $S^\sharp$ from the displacement equation $Z\widetilde{S} - \widetilde{S}Z^T = \widetilde{G}J\widetilde{G}^T$. Let $a^\sharp(x)$ and $b^\sharp(x)$ be the polynomials defined by the first and second columns of $\widetilde{G}$, respectively. Then, the Bezoutian generated by these two polynomials will be exactly $S^\sharp$.

Lemma 7 *If we perform Gaussian elimination on a Bezoutian, then the result will still be a Bezoutian.*

Proof First, it is easy to see that Gaussian elimination on the matrix $S^\sharp$ corresponds to Gaussian elimination on its generator. Second, if we perform Gaussian elimination on $S^\sharp$ and then pad the resultant matrix with a row and a column of zeros, the result will be the same as the result of padding $S^\sharp$ first to obtain $\widetilde{S}$ and performing the

same steps of Gaussian elimination on $\widetilde{S}$. This is because the corresponding steps of Gaussian elimination will not alter a column or a row of zeros. Therefore, if we have an arbitrary Bezoutian $S^{\sharp}$, we know that $\widetilde{S}$ has displacement rank 2. Let us say Gaussian elimination is performed on $S^{\sharp}$ to obtain a different matrix $S^{(1)}$. From the discussion above, we can infer that the same steps of Gaussian elimination performed on $\widetilde{S}$ will result in $\begin{bmatrix} S^{(1)} & 0 \\ 0 & 0 \end{bmatrix}$. Let $\widetilde{G}$ be the generator of $\widetilde{S}$ and $G^{(1)}$ be the result after applying steps of Gaussian elimination to $\widetilde{G}$. It is clear that $G^{(1)}$ is the generator of $\begin{bmatrix} S^{(1)} & 0 \\ 0 & 0 \end{bmatrix}$, which in turn implies $S^{(1)}$ is a Bezoutian, hence the result.

The above result immediately implies the following statement.

Corollary 5 *The Schur complement of a Bezoutian is a Bezoutian.*

Proof This follows because Schur complementation is equivalent to Gaussian elimination.

4 Schur–Euclid-Type Algorithm via Bezoutian Having Hessenberg Structure

This section describes Hessenberg displacement structure of the Bezoutian associated with reverse polynomials expanded in a monomial $P = \{1, x, \cdots, x^n\}$ basis and then generalizes it to the basis $Q = \{Q_0, Q_1, \cdots, Q_n\}$ where deg $Q_k(x) = k$. The main idea is to explore the transformation of Hessenberg displacement structure of a Bezoutian from a monomial basis P and to the generalized basis Q. Once it is established we see the connection of the Schur complement of a Bezoutian to the Schur–Euclid–Hessenberg algorithm via generator updates of the generalized Bezoutian and confederate matrix.

4.1 Hessenberg Displacement Structure of Bezoutian Over Monomial Basis

Here we use the displacement equation of a Bezoutian (14) associated with reverse polynomials over a monomial basis to address the connection among generator updates of the companion matrix, the Schur complement of the Bezoutian, and polynomial division.

Definition 5 A matrix R_1 is said to have Hessenberg displacement structure if it satisfies

$$F_1 R_1 - R_1 A_1 = G^{(1)} B^{(1)} \tag{17}$$

where A_1 is an upper Hessenberg matrix and F_1 is a lower Hessenberg Matrix.

In Lemma 5, we have seen the connection of the Bezoutian $S^\sharp = Bez_P(a^\sharp, b^\sharp)$ to the displacement equation $C_a^T S^\sharp - S^\sharp C_a = 0$ and vice-versa. Since C_a is an upper Hessenberg matrix, by following the Definition 5, we can say that the Bezoutian has the Hessenberg displacement structure associated with reverse polynomial over a monomial basis.

In the following, we use the Hessenberg displacement structure of a Bezoutian over a monomial basis to see a connection to generator updates of the companion matrix to polynomial division.

Lemma 8 *Let C_a (15) be the companion matrix of the polynomial $a(x)$ satisfying $C_a^T S^\sharp - S^\sharp C_a = 0$ where $S^\sharp = Bez_P(a^\sharp, b^\sharp)$ and deg $a(x) >$ deg $b(x)$. Then the generator update of C_a is the companion matrix of the polynomial $b(x)$.*

Proof Let deg $a(x) = n$ and deg $b(x) = n - k$. Since the Bezoutian satisfies the Hessenberg displacement structure $C_a^T S^\sharp - S^\sharp C_a = 0$ one can apply Lemma 3 to update the generators. Thus the updated companion matrix results in:

$$C_{new} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix} - \begin{bmatrix} 0 & \cdots & 0 & 1 \\ 0 & \cdots & 0 & 0 \\ \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & 0 \end{bmatrix} \cdot R_{11}^{-1} \cdot R_{12} \tag{18}$$

However, from another displacement equation of the Bezoutian, i.e., following the formula (13) one can state:

$$ZR - RZ^T = G^{(1)} \bar{J} G^{(1)^T} \tag{19}$$

where $R = \begin{bmatrix} Bez(b^\sharp, a^\sharp) & 0 \\ 0 & 0 \end{bmatrix} = -\begin{bmatrix} S^\sharp & 0 \\ 0 & 0 \end{bmatrix}$, $\bar{J} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$, Z is the lower shift matrix, and $G^{(1)} = \begin{bmatrix} a_n & 0 \\ \vdots & \vdots \\ a_{n-k+1} & 0 \\ a_{n-k} & b_{n-k} \\ \vdots & \vdots \\ a_0 & b_0 \end{bmatrix}$.

With the help of the above equation one can obtain the following equations by considering the first k columns of R:

$$\left[Zr_1 |\ Zr_2 - r_1 |\ Zr_3 - r_2 |\ \ldots\ | Zr_k - r_{k-1} \right] = \left[a_n \mathbf{b} |\ a_{n-1}\mathbf{b} |\ a_{n-2}\mathbf{b} |\ \ldots\ | a_{n-k+1}\mathbf{b} \right] \tag{20}$$

From these it is possible to obtain the following relations:

$$\begin{aligned}
Zr_1 &= a_n \mathbf{b} \\
Zr_2 &= r_1 + a_{n-1}\mathbf{b} = Z^T a_n \mathbf{b} + a_{n-1}\mathbf{b} \\
Zr_3 &= r_2 + a_{n-2}\mathbf{b} = (Z^T)^2 a_n \mathbf{b} + Z^T a_{n-1}\mathbf{b} + a_{n-2}\mathbf{b} \\
&\vdots \\
Zr_k &= r_{k-1} + a_{n-k+1}\mathbf{b} = (Z^T)^{k-1} a_n \mathbf{b} + (Z^T)^{k-2} a_{n-1}\mathbf{b} + \ldots + Z^T a_{n-k+2}\mathbf{b} + a_{n-k+1}\mathbf{b}
\end{aligned}$$

Let R_{11} be the $k \times k$ upper block of R. Then, from the above equations it is possible to express R_{11} as:

$$\begin{bmatrix}
0 & & \cdots & 0 & a_n b_{n-k} \\
0 & & \cdot & a_n b_{n-k} & a_n b_{n-k-1} + a_{n-1} b_{n-k} \\
\vdots & \cdot & & & \vdots \\
0 & a_n b_{n-k} & & & \\
a_n b_{n-k} & a_n b_{n-k-1} + a_{n-1} b_{n-k} & \cdots & & a_n b_{n-(2k-1)} + a_{n-1} b_{n-(2k)} + \ldots + a_{n-k+1} b_{n-k}.
\end{bmatrix}$$

Let $\widetilde{I}$ be the antidiagonal matrix. Multiplying the above (R_{11}) by $\widetilde{I}$ from the right:

$$\begin{bmatrix}
a_n b_{n-k} & 0 & \cdots & 0 & 0 \\
a_n b_{n-k-1} + a_{n-1} b_{n-k} & a_n b_{n-k} & 0 & \vdots & 0 \\
\vdots & & \cdot & 0 & \vdots \\
 & & & a_n b_{n-k} & 0 \\
a_n b_{n-(2k-1)} + a_{n-1} b_{n-2k} + \ldots + a_{n-k+1} b_{n-k} & & \cdots & a_n b_{n-k-1} + a_{n-1} b_{n-k} & a_n b_{n-k}
\end{bmatrix}$$

However this can easily be factored as:

$$R_{11} \cdot \widetilde{I} = \begin{bmatrix}
b_{n-k} & & & \\
b_{n-k-1} & b_{n-k} & & \\
\vdots & b_{n-k-1} & \ddots & \\
 & & \ddots & \ddots \\
b_{n-(2k-1)} & & \cdots & b_{n-k-1}\ b_{n-k}
\end{bmatrix} \cdot \begin{bmatrix}
a_n & & & \\
a_{n-1} & a_n & & \\
\vdots & a_{n-1} & \ddots & \\
 & & \ddots & \ddots \\
a_{n-k+1} & & \cdots & a_{n-1}\ a_n
\end{bmatrix} \tag{21}$$

So, $R_{11} \cdot \widetilde{I} = B_k A_k$ where we denote the Toeplitz matrix composed of the coefficients of $a(x)$ on the lower diagonals in the above Eq. (21) as A_k and similarly for B_k. Therefore, $R_{11}^{-1} = \widetilde{I} A_k^{-1} B_k^{-1}$.

Let R_{21} be the $(n-k+1) \times k$ block of R right below R_{11}, i.e.,

$$R_{21} = \begin{bmatrix} a_n b_{n-k-1} & a_n b_{n-k-2} + a_{n-1} b_{n-k-1} & \dots & a_n b_{n-2k} + a_{n-1} b_{n-2k+1} + \dots + a_{n-k+1} b_{n-k-1} \\ a_n b_{n-k-2} & a_n b_{n-k-3} + a_{n-1} b_{n-k-2} & \dots & \vdots \\ \vdots & \vdots & & \\ a_n b_2 & a_n b_1 + a_{n-1} b_2 & \dots & a_{n-k+3} b_0 + a_{n-k+2} b_1 + a_{n-k+1} b_2 \\ a_n b_1 & a_n b_0 + a_{n-1} b_1 & \dots & a_{n-k+2} b_0 + a_{n-k+1} b_1 \\ a_n b_0 & a_{n-1} b_0 & \dots & a_{n-k+1} b_0 \\ 0 & 0 & \dots & 0 \end{bmatrix}$$

Let us compute $R_{21} \cdot \widetilde{I}$:

$$R_{21} \cdot \widetilde{I} = \begin{bmatrix} a_n b_{n-2k} + a_{n-1} b_{n-2k+1} + \dots + a_{n-k+1} b_{n-k-1} & \dots & a_n b_{n-k-2} + a_{n-1} b_{n-k-1} & a_n b_{n-k-1} \\ \vdots & \dots & a_n b_{n-k-3} + a_{n-1} b_{n-k-2} & a_n b_{n-k-2} \\ & & \vdots & \vdots \\ a_{n-k+3} b_0 + a_{n-k+2} b_1 + a_{n-k+1} b_2 & \dots & a_n b_1 + a_{n-1} b_2 & a_n b_2 \\ a_{n-k+2} b_0 + a_{n-k+1} b_1 & \dots & a_n b_0 + a_{n-1} b_1 & a_n b_1 \\ a_{n-k+1} b_0 & \dots & a_{n-1} b_0 & a_n b_0 \\ 0 & \dots & 0 & 0 \end{bmatrix}$$

It is possible to factor the above as follows:

$$R_{21} \cdot \widetilde{I} = \begin{bmatrix} b_{n-2k} & b_{n-2k+1} & b_{n-2k+2} & \dots & b_{n-k-1} \\ b_{n-2k-1} & b_{n-2k} & b_{n-2k+1} & \dots & b_{n-k-2} \\ \vdots & & & & \\ b_0 & & & & \vdots \\ 0 & \ddots & & & \\ & \ddots & & & \vdots \\ \vdots & & & \ddots & \\ & & & \ddots & b_0 \\ 0 & & \dots & & 0 \end{bmatrix} \cdot \begin{bmatrix} a_n & & & \\ a_{n-1} & a_n & & \\ \vdots & a_{n-1} & \ddots & \\ & & \ddots & \ddots \\ a_{n-k+1} & & \dots & a_{n-1} \; a_n \end{bmatrix}$$

From this it follows that $R_{21} = \widehat{B}_k \cdot A_k \cdot \widetilde{I}$ where $\widehat{B}_k$ is the first matrix in the RHS of the above equation. Therefore

$$R_{21} R_{11}^{-1} = (\widehat{B}_k A_k \widetilde{I}) \cdot (\widetilde{I} A_k^{-1} B_k^{-1}) = \widehat{B}_k \cdot B_k^{-1}.$$

Moreover one can say that $(R_{11}^{-1} R_{12})^T = R_{21} R_{11}^{-1}$. Thus

$$(R_{11}^{-1}R_{12})^T = \begin{bmatrix} b_{n-2k} & b_{n-2k+1} & \cdots & & b_{n-k-1} \\ b_{n-2k-1} & b_{n-2k} & \cdots & & b_{n-k-2} \\ b_{n-2k-2} & b_{n-2k-1} & \cdots & & b_{n-k-3} \\ \vdots & & & & \\ & & & & \vdots \\ b_0 & & & & \\ 0 & \ddots & & & \\ & \ddots & & & \\ \vdots & & & \ddots & \\ & & & \ddots & b_0 \\ 0 & & \cdots & & 0 \end{bmatrix} \cdot \begin{bmatrix} b_{n-k} & & & & \\ b_{n-k-1} & b_{n-k} & & & \\ b_{n-k-2} & b_{n-k-1} & b_{n-k} & & \\ \vdots & & & \ddots & \\ b_{n-(2k-1)} & b_{n-(2k-2)} & \cdots & b_{n-k-1} & b_{n-k} \end{bmatrix}^{-1} \tag{22}$$

Since the generator update of C_a in Eq. (18) uses only the last row of $R_{11}^{-1}R_{12}$, one has to consider only the last column of its transpose (after peeling off the last entry) which is $\begin{bmatrix} \frac{b_{n-k-1}}{b_{n-k}} \\ \frac{b_{n-k-2}}{b_{n-k}} \\ \frac{b_{n-k-3}}{b_{n-k}} \\ \vdots \\ \frac{b_0}{b_{n-k}} \end{bmatrix}$. Thus the updated matrix C_{new} in (18) is given by:

$$\begin{bmatrix} -\frac{b_{n-k-1}}{b_{n-k}} & -\frac{b_{n-k-2}}{b_{n-k}} & \cdots & -\frac{b_1}{b_{n-k}} & -\frac{b_0}{b_{n-k}} \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \ddots & \ddots & & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix}$$

which is the companion matrix for $b(x)$ and hence the result.

Corollary 6 *The first column of the Bezoutian $S^\sharp = Bez_P(a^\sharp, b^\sharp)$ where deg $a(x) >$ deg $b(x)$ contains scalar multiples of the coefficients of $b(x)$.*

Proof This result is trivial as the first column of R, i.e., $r_1 = Z^T a_n \mathbf{b}$ where $\mathbf{b} = \begin{bmatrix} b_{n-k} & b_{n-k-1} & \cdots & b_0 \end{bmatrix}^T$ and $R = -\begin{bmatrix} S^\sharp & 0 \\ 0 & 0 \end{bmatrix}$.

The next result shows the updated first column of the Schur complement of a Bezoutian in the kth step.

Corollary 7 *The first column of the Schur complement of $S^\sharp = Bez_P(a^\sharp, b^\sharp)$ where deg $a(x) >$ deg $b(x)$, contains scalar multiples of the coefficients of the polynomial $-c(x)$ which is the remainder of the polynomial division of $a(x)$ by $b(x)$.*

Proof Let us partition the generator $G^{(1)}$ in (19) as $G^{(1)} = \begin{bmatrix} G_1 \\ G_2 \end{bmatrix}$ where $G_1 = \begin{bmatrix} a_n & 0 \\ a_{n-1} & 0 \\ \vdots & \vdots \\ a_{n-k+1} & 0 \end{bmatrix}$ and $G_2 = \begin{bmatrix} a_{n-k} & b_{n-k} \\ a_{n-k-1} & b_{n-k-1} \\ \vdots & \vdots \\ a_0 & b_0 \end{bmatrix}$. Since $S^\sharp$ satisfies the displacement equation (19) with lower shift matrix Z, one can apply the block form of Lemma 2 to update the generator $G^{(1)}$ via:

$$\begin{bmatrix} d_{n-k} & b_{n-k} \\ d_{n-k-1} & b_{n-k-1} \\ \vdots & \vdots \\ d_0 & b_0 \end{bmatrix} = \begin{bmatrix} a_{n-k} & b_{n-k} \\ a_{n-k-1} & b_{n-k-1} \\ \vdots & \vdots \\ a_0 & b_0 \end{bmatrix} - R_{21}R_{11}^{-1} \begin{bmatrix} a_n & 0 \\ a_{n-1} & 0 \\ \vdots & \vdots \\ a_{n-k+1} & 0 \end{bmatrix}$$

which results in the formula:

$$\begin{bmatrix} d_{n-k} \\ d_{n-k-1} \\ \vdots \\ d_0 \end{bmatrix} = \begin{bmatrix} a_{n-k} \\ a_{n-k-1} \\ \vdots \\ a_0 \end{bmatrix} - R_{21}R_{11}^{-1} \begin{bmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_{n-k+1} \end{bmatrix}$$

The matrices R_{11} and R_{21} in the above system are expressed explicitly in Lemma 8 and are the same as the matrices B_k and $\widehat{B}_k$ respectively defined in Lemma 1. Thus combining both Lemmas results in:

$$\begin{bmatrix} d_{n-k} \\ d_{n-k-1} \\ d_{n-k-2} \\ \vdots \\ d_0 \end{bmatrix} = - \begin{bmatrix} 0 \\ c_{n-k-1} \\ c_{n-k-2} \\ \vdots \\ c_0 \end{bmatrix} + q_0 \begin{bmatrix} b_{n-k} \\ b_{n-k-1} \\ b_{n-k-2} \\ \vdots \\ b_0 \end{bmatrix}$$

where from Lemma 1, $-c(x) = -\sum_{i=k+1}^{n} c_{n-i}x^{n-i}$ is the remainder and q_0 is the constant term in the quotient of the polynomial division of $a(x)$ by $b(x)$. Thus the generator update of $G^{(1)}$ via the Bezoutian $Bez_P(a^\sharp, b^\sharp)$ corresponds to the polynomial division of $a(x)$ by $b(x)$. Moreover following Lemma 2, the Schur complement of the Bezoutian, which is the new Bezoutian $Bez_P(b^\sharp, c^\sharp)$, also satisfies the displacement equation (19) with the above generator updates. Thus, as in Lemma 8, one can recover the scalar multiple of the coefficients of $c(x)$ from the first column of $Bez_P(b^\sharp, c^\sharp)$.

Theorem 1 *Let the Bezoutian $S^\sharp = Bez_P(a^\sharp, b^\sharp)$ where deg $a(x) >$ deg $b(x)$ and C_a is the companion matrix defined via* (15)*. Then the generator update of the Bezoutian*

$S^\sharp$ over the displacement equation $C_a^T S^\sharp - S^\sharp C_a = 0$ coincides with the polynomial division of $a(x)$ by $b(x)$.

Proof Lemma 5 shows that the Bezoutian $S^\sharp$ satisfies $C_a^T S^\sharp - S^\sharp C_a = 0$. Since C_a has upper Hessenberg structure one can apply Lemma 3 to update $S^\sharp$, C_a and C_a^T. Here there is no need to update $G^{(1)}$ and $B^{(1)}$ since they are both 0. Moreover, updates for C_a and C_a^T via Lemma 3 will preserve the upper and lower Hessenberg structures. As we know the Bezoutian $S^\sharp$ is completely determined by polynomials $a^\sharp(x)$ and $b^\sharp(x)$. By Lemma 8 the polynomial $b(x)$ can be recovered from the generator update of the companion matrix C_a, i.e., after generator updates the companion matrix of the polynomial $a(x)$ becomes the companion matrix of the polynomial $b(x)$. Now by Corollary 7, one can recover the scalar multiple of the coefficients of $-c(x)$ which is the remainder of the polynomial division of $a(x)$ by $b(x)$ via the first column of the Schur complement of the Bezoutian $Bez_P(b^\sharp, c^\sharp)$. Hence the result.

The next section generalizes the result in this section having polynomials expanded over the basis $\{Q_k(x)\}_{k=0}^n$ where $\deg Q_k(x) = k$.

4.2 Hessenberg Displacement Structure of Bezoutian Over Generalized Basis

In this section, we first generalize Hessenberg displacement structure of a Bezoutian over monomial basis $P = \{x^k\}_{k=0}^n$ to an arbitrary basis $Q = \{Q_k(x)\}_{k=0}^n$ where $\deg Q_k(x) = k$. As a result of this, we will have a new displacement equation with the generalized Bezoutian and confederate matrix. Next we elaborate the Schur complement of the generalized Bezoutian over $\{Q\}$ and use this to analyze the generator updates of a generalized Bezoutian with the polynomial division over $\{Q\}$. Finally, we state the Schur–Euclid–Hessenberg algorithm.

Definition 6 Let $\{Q\} = \{Q_0(x), Q_1(x), \cdots, Q_n(x)\}$ be a system of polynomials satisfying $\deg Q_k(x) = k$ and, $a(x)$ and $b(x)$ be polynomials of degree not greater than n. Then a matrix $S_Q = [\hat{s}_{ij}]$ is the generalized Bezoutian associated with the reverse polynomials $a^\sharp(x)$ and $b^\sharp(x)$ over $\{Q\}$ say $S_Q = Bez_Q(a^\sharp, b^\sharp)$ if

$$\begin{aligned}\frac{a^\sharp(x)\cdot b^\sharp(y) - b^\sharp(x)\cdot a^\sharp(y)}{x-y} &= \sum_{i,j=0}^{n-1} \hat{s}_{ij} Q_i(x) Q_j(y) \\ &= \begin{bmatrix} Q_0(x) & Q_1(x) & \cdots & Q_{n-1}(x) \end{bmatrix} S_Q \begin{bmatrix} Q_0(y) \\ Q_1(y) \\ \vdots \\ Q_{n-1}(y) \end{bmatrix} \end{aligned} \quad (23)$$

The next result shows a relationship between a Bezoutian for polynomials over the monomial basis $\{P\}$ and the generalized basis $\{Q\} = \{Q_0(x), Q_1(x), \cdots, Q_n(x)\}$ having deg $Q_k(x) = k$.

Lemma 9 *Let B_{PQ} be uni upper triangular basis transformation matrix corresponding to passing basis $\{P\} = \{x^k\}_{k=0}^n$ to basis $\{Q\} = \{Q_k(x)\}_{k=0}^n$ where* deg $Q_k(x) = k$ *via:*

$$\left[Q_0(x)\ Q_1(x)\ \cdots\ Q_{n-1}(x)\right] B_{PQ} = \left[1\ x\ \cdots\ x^{n-1}\right].$$

Then

$$S_Q = B_{PQ}\, S^\sharp\, B_{PQ}^T \tag{24}$$

where $S^\sharp = Bez_P(a^\sharp, b^\sharp)$ and $S_Q = Bez_Q(a^\sharp, b^\sharp)$.

Proof Recall from the Definition 4 of the Bezoutian associated with the reverse polynomials over basis $\{P\}$

$$\frac{a^\sharp(x)\cdot b^\sharp(y) - b^\sharp(x)\cdot a^\sharp(y)}{x-y} = \left[1\ x\ x^2\ \cdots\ x^{n-1}\right] S^\sharp \begin{bmatrix} 1 \\ y \\ y^2 \\ \vdots \\ y^{n-1} \end{bmatrix}$$

We can revise the RHS of the above system:

$$\begin{aligned} \frac{a^\sharp(x)\cdot b^\sharp(y) - b^\sharp(x)\cdot a^\sharp(y)}{x-y} &= \left[\,1\ x\ \cdots\ x^{n-1}\,\right] B_{PQ}^{-1}\, B_{PQ}\, S^\sharp\, B_{PQ}^T\, B_{PQ}^{-T} \begin{bmatrix} 1 \\ y \\ \vdots \\ y^{n-1} \end{bmatrix} \\ &= \left[\,Q_0(x)\ Q_1(x)\ \cdots\ Q_{n-1}(x)\,\right] B_{PQ}\, S^\sharp\, B_{PQ}^T \begin{bmatrix} Q_0(y) \\ Q_1(y) \\ \vdots \\ Q_{n-1}(y) \end{bmatrix}. \end{aligned}$$

Following Definition 6 gives the result.

To generalize the displacement equation for a Bezoutian we have to explore the Hessenberg structured confederate matrix. Thus we will give the definition of a confederate matrix introduced in [37] next.

Definition 7 Let polynomials $\{Q\} = \{Q_0(x), Q_1(x), Q_2(x), ..., Q_n(x)\}$ with deg $Q_k(x) = k$ be specified by the recurrence relation

$$Q_k(x) = \alpha_k x Q_{k-1}(x) - r_{k-1,k} Q_{k-1}(x) - r_{k-2,k} Q_{k-2}(x) - \ldots - r_{0,k} Q_0(x), \alpha_k \neq 0$$

for $k > 0$ and $Q_0(x)$ is a constant. Define for the polynomial

$$a(x) = a_0 Q_0(x) + a_1 Q_1(x) + \ldots + a_n Q_n(x)$$

its confederate matrix (with respect to the polynomial system Q) by

$$C_Q(a) = \begin{bmatrix} \frac{r_{0,1}}{\alpha_1} & \frac{r_{0,2}}{\alpha_2} & \frac{r_{0,3}}{\alpha_3} & \ldots & \frac{r_{0,n}}{\alpha_n} - \frac{a_0}{\alpha_n a_n} \\ \frac{1}{\alpha_1} & \frac{r_{1,2}}{\alpha_2} & \frac{r_{1,3}}{\alpha_3} & \ldots & \frac{r_{1,n}}{\alpha_n} - \frac{a_1}{\alpha_n a_n} \\ 0 & \frac{1}{\alpha_2} & \frac{r_{2,3}}{\alpha_3} & \ldots & \frac{r_{2,n}}{\alpha_n} - \frac{a_2}{\alpha_n a_n} \\ \vdots & \vdots & \ddots & & \vdots \\ 0 & 0 & \cdots & \frac{1}{\alpha_{n-1}} & \frac{r_{n-1,n}}{\alpha_n} - \frac{a_{n-1}}{\alpha_n a_n}. \end{bmatrix}$$

In the special case where $a(x) = Q_n(x)$, we have $a_0 = a_1 = \cdots = a_{n-1} = 0$. We refer to [37] for many useful properties of the confederate matrix and only recall here that

$$Q_k(x) = \alpha_0 \alpha_1 \cdots \alpha_k \cdot det(xI - [C_Q(a)]_{k \times k}),$$

and

$$a(x) = \alpha_0 \alpha_1 \cdots \alpha_n a_n \cdot det(xI - [C_Q(a)]),$$

where $[C_Q(a)]_{k \times k}$ denotes the $k \times k$ leading submatrix of $C_Q(a)$.

As we have seen the generalized Bezoutian associated with reverse polynomials and confederate matrix capturing recurrence relations over $\{Q\}$, we will next generalize the displacement equation $C_a^T S^\sharp - S^\sharp C_a = 0$ passing $Bez_P(a^\sharp, b^\sharp)$ to the generalized Bezoutian $S_Q = Bez_Q(a^\sharp, b^\sharp)$ and companion matrix C_a to the confederate matrix C_Q.

Theorem 2 *Let* $\{Q\} = \{Q_0(x), Q_1(x), Q_2(x), ..., Q_n(x)\}$ *where* $\deg Q_k(x) = k$ *be the system of polynomials satisfying recurrence relations*

$$\begin{aligned} Q_0(x) &= 1 \\ Q_k(x) &= xQ_{k-1}(x) - r_{k-1,k}Q_{k-1}(x) - r_{k-2,k}Q_{k-2}(x) - \ldots - r_{0,k}Q_0(x) \end{aligned} \quad (25)$$

$a(x) = a_0 Q_0(x) + a_1 Q_1(x) + \ldots + a_n Q_n(x)$ *and similarly for* $b(x)$. *Then a matrix* $S_Q = Bez_Q(a^\sharp, b^\sharp)$ *is a Bezoutian associated with reverse polynomials* $a^\sharp(x)$ *and* $b^\sharp(x)$ *if and only if* S_Q *satisfies the equation*

$$C_Q^T S_Q - S_Q C_Q = 0 \quad (26)$$

for some confederate matrix

$$C_Q = \tilde{I} C_{Q'}^T \tilde{I} \quad (27)$$

where

$$C_{Q'} = \begin{bmatrix} r_{0,1} & r_{0,2} & r_{0,3} & \cdots & r_{0,n} - \frac{a_0}{a_n} \\ 1 & r_{1,2} & r_{1,3} & \cdots & r_{1,n} - \frac{a_1}{a_n} \\ 0 & 1 & r_{2,3} & \cdots & r_{2,n} - \frac{a_2}{a_n} \\ \vdots & \vdots & \ddots & & \vdots \\ 0 & 0 & \cdots & 1 & r_{n-1,n} - \frac{a_{n-1}}{a_n}. \end{bmatrix} \tag{28}$$

Proof In the monomial basis $\{P\}$ it has been proven that

$$C_a^T S^\sharp - S^\sharp C_a = 0$$

where $S^\sharp = Bez_P(a^\sharp, b^\sharp)$ and $C_a = \begin{bmatrix} -\frac{a_{n-1}}{a_n} & -\frac{a_{n-2}}{a_n} & -\frac{a_{n-3}}{a_n} & \cdots & -\frac{a_0}{a_n} \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}$. Thus it is possible to rewrite the above system as

$$S_1 C_1^T - C_1 S_1 = 0 \tag{29}$$

where $S_1 = \tilde{I} S^\sharp \tilde{I} = Bez_P(a, b)$ and $C_1 = \tilde{I} C_a^T \tilde{I}$. Now with the help of the structure of the uni upper triangular basis transformation matrix B_{PQ} together with the result [32] one can revise the above system as

$$S_{Q'} C_{Q'}^T - C_{Q'} S_{Q'} = 0$$

where $S_{Q'} = B_{PQ} S_1 B_{PQ}^T = Bez_Q(a, b)$ and $C_{Q'} = B_{PQ} C_1 B_{PQ}^{-1}$ is the confederate matrix given by (28). By rearranging the above system we get

$$(\tilde{I} S_{Q'} \tilde{I})(\tilde{I} C_{Q'}^T \tilde{I}) - (\tilde{I} C_{Q'} \tilde{I})(\tilde{I} S_{Q'} \tilde{I}) = 0$$

yields the result:

$$C_Q^T S_Q - S_Q C_Q = 0.$$

Conversely if $C_Q^T S_Q - S_Q C_Q = 0$ and $S_Q = [\hat{s}_{ij}]$ then the second column of S_Q is given by:

$$C_Q^T \hat{s}_{i,1} - \left(r_{n-1,n} - \frac{a_{n-1}}{a_n} \right) \hat{s}_{i,1} - \hat{s}_{i,2} = 0 \Rightarrow \hat{s}_{i,2} = C_Q^T \hat{s}_{i,1} - \left(r_{n-1,n} - \frac{a_{n-1}}{a_n} \right) \hat{s}_{i,1}$$

the third column of S_Q is given by:

$$C_Q^T \hat{s}_{i,2} - \left(r_{n-2,n} - \frac{a_{n-2}}{a_n} \right) \hat{s}_{i,1} - r_{n-2,n-1} \hat{s}_{i,2} - \hat{s}_{i,3} = 0$$

$$\hat{s}_{i,3} = C_Q^T \hat{s}_{i,2} - \left(r_{n-2,\,n} - \frac{a_{n-2}}{a_n} \right) \hat{s}_{i,1} - r_{n-2,\,n-1}\, \hat{s}_{i,2}$$

proceeding recursively the kth column of S_Q is given by:

$$\hat{s}_{i,k} = C_Q^T \hat{s}_{i,k-1} - \left(r_{n-k+1,\,n} - \frac{a_{n-k+1}}{a_n} \right) \hat{s}_{i,1} - r_{n-k+1,\,n-1}\, \hat{s}_{i,2} - \cdots - r_{n-k+1,\,n-k+2}\, \hat{s}_{i,k-1}.$$

Thus one can recover all columns of S_Q and it should be clear that since S_Q satisfies (26) there is no other matrix which satisfies (26) and has the same first column $\hat{s}_{i,1}$. Thus $S_Q = Bez_Q(a^\sharp, b^\sharp)$. □

It has been shown in Lemma 7 and Corollary 5 that the Schur complement of a Bezoutian is Bezoutian. Thus in the following result, we will generalize the result obtained in Sect. 3. We show that the Schur complement of a Bezoutian S_Q with respect to the generalized basis $\{Q\}$ is congruent to the Schur complement of the Bezoutian $S^\sharp$ with respect to the monomial basis $\{P\}$.

Theorem 3 *The Schur complement of the generalized Bezoutian S_Q over the basis $\{Q\} = \{Q_k(x)\}_{k=0}^n$ where deg $Q_k(x) = k$ is congruent to the Schur complement of the Bezoutian $S^\sharp$ over monomials $\{P\} = \{x^k\}_{k=0}^n$.*

Proof Let us partition the Bezoutian matrix: $S^\sharp = \begin{bmatrix} S_{11} & s_{12} \\ s_{12}^T & s_{22} \end{bmatrix}$. From Lemma 9, we know that $S_Q = B_{PQ}\, S^\sharp\, B_{PQ}^T$ so the basis transformation matrix which is an upper triangular matrix having 1's along the diagonal can be partitioned as $B_{PQ} = \begin{bmatrix} U_{11} & u_{12} \\ 0 & 1 \end{bmatrix}$. Now analyze the block products of $S_Q = B_{PQ}\, S^\sharp\, B_{PQ}^T$ to find its Schur complement.

$$\begin{aligned} S_Q &= \begin{bmatrix} U_{11} & u_{12} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} S_{11} & s_{12} \\ s_{12}^T & s_{22} \end{bmatrix} \begin{bmatrix} U_{11}^T & 0 \\ u_{12}^T & 1 \end{bmatrix} \\ &= \begin{bmatrix} U_{11} & u_{12} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I & \frac{1}{s_{22}} s_{12} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} S_{11} - \frac{1}{s_{22}} s_{12} s_{12}^T & 0 \\ 0 & s_{22} \end{bmatrix} \begin{bmatrix} I & 0 \\ \frac{1}{s_{22}} s_{12}^T & 1 \end{bmatrix} \begin{bmatrix} U_{11}^T & 0 \\ u_{12}^T & 1 \end{bmatrix} \\ &= \begin{bmatrix} * & s_{12}u_{11} + s_{22}u_{12} \\ s_{12}^T u_{11}^T + s_{22} u_{12}^T & s_{22} \end{bmatrix} \end{aligned}$$

where $* := U_{11} \left(S_{11} - \frac{1}{s_{22}} s_{12} s_{12}^T \right) U_{11}^T + (s_{12}u_{11} + s_{22}u_{12}) \left(\frac{1}{s_{22}} s_{12}^T u_{11}^T + u_{12}^T \right)$. Thus Schur complement of S_Q say S_{Qs} is given by:

$$\begin{aligned} S_{Qs} &= \left(U_{11} \left(S_{11} - \frac{1}{s_{22}} s_{12} s_{12}^T \right) U_{11}^T + (s_{12}u_{11} + s_{22}u_{12}) \left(\frac{1}{s_{22}} s_{12}^T u_{11}^T + u_{12}^T \right) \right) \\ &\quad - \frac{1}{s_{22}} (s_{12}u_{11} + s_{22}u_{12}) \left(s_{12}^T u_{11}^T + s_{22} u_{12}^T \right) \\ &= U_{11} \left(S_{11} - \frac{1}{s_{22}} s_{12} s_{12}^T \right) U_{11}^T \end{aligned}$$

Hence the result.

The next result shows the connection of generator updates of the generalized Bezoutian to polynomial division over basis $\{Q\}$.

Theorem 4 *Let* $\{Q\} = \{Q_0, Q_1, Q_2, ..., Q_n\}$ *where deg* $Q_k(x) = k$ *be the system of polynomials satisfying recurrence relations* (25) *and* $S_Q = Bez_Q(a^\sharp, b^\sharp) = [\hat{s}_{ij}]$. *If* $a(x) = a_0Q_0(x) + a_1Q_1(x) + \ldots + a_nQ_n(x)$ *and* $b(x) = b_0Q_0(x) + b_1Q_1(x) + \ldots + b_{n-k}Q_{n-k}(x)$ *then the coefficients of the remainder* $-c(x)$ *of the polynomial division* $a(x)$ *by* $b(x)$ *can be recovered from:*

$$-\begin{bmatrix} c_{n-k-1} \\ c_{n-k-2} \\ \vdots \\ c_0 \end{bmatrix} = \left[\left[C_Q^T - \left(r_{n-1,\,n} - \frac{a_{n-1}}{a_n}\right) I_n\right] \hat{s}_{i,1}\right]' - \frac{\hat{s}_{12}}{\hat{s}_{11}} \left[\hat{s}_{i,1}\right]' \tag{30}$$

where

$$C_Q = \begin{bmatrix} r_{n-1,\,n} - \frac{a_{n-1}}{a_n} & r_{n-2,\,n} - \frac{a_{n-2}}{a_n} & r_{n-3,\,n} - \frac{a_{n-3}}{a_n} & \cdots & r_{0,\,n} - \frac{a_0}{a_n} \\ 1 & r_{n-2,\,n-1} & r_{n-3,\,n-1} & \cdots & r_{0,\,n-1} \\ 0 & 1 & r_{n-3,\,n-2} & \cdots & r_{0,\,n-2} \\ \vdots & \ddots & \ddots & & \\ 0 & \cdots & 0 & 1 & r_{0,\,1} \end{bmatrix} \tag{31}$$

and prime means peeling off the first k components.

Proof We have shown in Theorem 2 that $C_Q^T S_Q - S_Q C_Q = 0$ and it is the same as $S_Q C_Q - C_Q^T S_Q = 0$. One can clearly see that C_Q is an upper Hessenberg matrix, so with that said, S_Q has Hessenberg displacement structure. Hence we can apply Lemma 3 to update C_Q. As deg $b(x)$ is $n-k$ let us partition matrices: $C_Q = \begin{bmatrix} C_q(k,k) & C_q(k,n-k) \\ C_q(n-k,k) & C_q(n-k,n-k) \end{bmatrix}$ and $S_Q = \begin{bmatrix} S_q(k,k) & S_q(k,n-k) \\ S_q(n-k,k) & S_q(n-k,n-k) \end{bmatrix}$ where $S_q(n-k,k) = \left[S_q(k,n-k)\right]^T$ and apply the block form of Lemma 3

$$\begin{aligned} C_{new} &= C_q(n-k,n-k) - C_q(n-k,k)\left[S_q(k,k)\right]^{-1} S_q(k,n-k) \\ &= \begin{bmatrix} r_{n-k-1,\,n-k} & r_{n-k-2,\,n-k} & r_{n-k-3,\,n-k} & \cdots & r_{0,\,n-k} \\ 1 & r_{n-k-2,\,n-k-1} & r_{n-k-3,\,n-k-1} & \cdots & r_{0,\,n-k-1} \\ 0 & 1 & r_{n-k-3,\,n-k-2} & \cdots & r_{0,\,n-k-2} \\ \vdots & \ddots & \ddots & & \vdots \\ 0 & \cdots & 0 & 1 & r_{0,\,1} \end{bmatrix} \\ &\quad - \begin{bmatrix} 0 & \cdots & 0 & 1 \\ 0 & \cdots & 0 & 0 \\ \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & 0 \end{bmatrix} \left[S_q(k,k)\right]^{-1} S_q(k,n-k). \end{aligned}$$

Thus when updating C_Q only the first row of $C_q(n-k, n-k)$ changes with respect to the last row of the product $\left[S_q(k,k)\right]^{-1} S_q(k, n-k)$. Let us restate the (1,1) block of C_{new} and (1,1), (1,2) blocks of S_Q in terms of a basis transformation matrix which can be partitioned as $B_{PQ} = \begin{bmatrix} B_{pq}(k,k) & B_{pq}(k, n-k) \\ B_{pq}(n-k, k) & \widehat{B}_{PQ} \end{bmatrix}$. Thus the above system can be seen as:

$$C_{new} = \tilde{I} \begin{bmatrix} r_{0,1} & r_{0,2} & r_{0,3} & \cdots & r_{0,n-k} \\ 1 & r_{1,2} & r_{1,3} & \cdots & r_{1,n-k} \\ 0 & 1 & r_{2,3} & \cdots & r_{2,n-k} \\ \vdots & \ddots & \ddots & & \vdots \\ 0 & \cdots & 0 & 1 & r_{n-k-1,n-k} \end{bmatrix}^T \tilde{I}$$

$$- \begin{bmatrix} 0 & \cdots & 0 & 1 \\ 0 & \cdots & 0 & 0 \\ \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & 0 \end{bmatrix} \left[S_q(k,k)\right]^{-1} \left[B_{pq}(k,k)\right]^T \left[B_{pq}(k,k)\right]^{-T} S_q(k, n-k)$$

$$= \widehat{B}_{PQ} \left[\begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix} - \begin{bmatrix} 0 & \cdots & 0 & 1 \\ 0 & \cdots & 0 & 0 \\ \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & 0 \end{bmatrix} \cdot R_{11}^{-1} \cdot R_{12} \right] \left[\widehat{B}_{PQ}\right]^{-1}.$$

Hence going back to the block form of the Bezoutian (22) we get

$$C_{new} = \begin{bmatrix} r_{n-k-1,n-k} - \frac{b_{n-k-1}}{b_{n-k}} & r_{n-k-2,n-k} - \frac{b_{n-k-2}}{b_{n-k}} & r_{n-k-3,n-k} - \frac{b_{n-k-3}}{b_{n-k}} & \cdots & r_{0,n-k} - \frac{b_0}{b_{n-k}} \\ 1 & r_{n-k-2,n-k-1} & r_{n-k-3,n-k-1} & \cdots & r_{0,n-k-1} \\ 0 & 1 & r_{n-k-3,n-k-2} & \cdots & r_{0,n-k-2} \\ \vdots & \ddots & \ddots & & \\ 0 & \cdots & 0 & 1 & r_{0,1}. \end{bmatrix}$$

Thus when a generalized Bezoutian S_Q satisfies the displacement equation $C_Q^T S_Q - S_Q C_Q = 0$ with an upper Hessenberg matrix C_Q, the generator update of the confederate matrix C_Q corresponding to polynomial $a(x)$ over $\{Q\}$ results in a confederate matrix C_{new} (say C_{Qb}) and that corresponds to the polynomial $b(x) = b_0 Q_0(x) + b_1 Q_1(x) + \ldots + b_{n-k} Q_{n-k}(x)$.

Now following Lemma 3, one can see a new system with generator updates as

$$S_{Qs} C_{Qb} - C_{Qb}^T S_{Qs} = 0$$

where C_{Qb} is the confederate matrix of $b(x)$ over basis $\{Q\}$ and S_{Qs} is the Schur complement of S_Q. We have shown in Theorem 3 that the Schur complement of S_Q, which is S_{Qs}, is congruent to the Schur complement of $S^\sharp$, which is $S^{(1)}$, i.e.,

$$S_{Qs} = U_{11} S^{(1)} U_{11}^T \tag{32}$$

where U_{11} is the (1,1) block of the uni upper triangular basis transformation matrix B_{PQ}. Moreover from Corollary 7, we have shown that the coefficients of the remainder over monomials can be retrieved from the first column of the Schur complement of $S^\sharp$ which is $S^{(1)}$. This together with the system (32) tells us that coefficients of the remainder over basis $\{Q\}$ can be retrieved from the first column of the S_{Qs}, the Schur complement of S_Q.

Before computing the coefficients of the remainder over $\{Q\}$ let us observe the second column of S_Q. This can be seen directly following Theorem 2 so the second column of S_Q is given by:

$$\hat{s}_{i,2} = \left[C_Q^T - \left(r_{n-1,\,n} - \frac{a_{n-1}}{a_n} \right) \right] \hat{s}_{i,1}.$$

The coefficients of the remainder $-c(x)$ over $\{Q\}$ can be retrieved from the first column of the Schur complement $S_{Qs} = [\tilde{s}_{ij}]$ so those can be retrieved from:

$$s_{i,1} = \left[\hat{s}_{i,2} \right]' - \hat{s}_{12} \left[\frac{1}{\hat{s}_{11}} \hat{s}_{i,1} \right]'$$

where prime means peeling off the first k components. Hence the result.

Remark 1 The above Theorem further shows that the generator update of a Bezoutian S_Q satisfying displacement equation $C_Q^T S_Q - S_Q C_Q = 0$ coincides with polynomial division over basis $\{Q\}$.

As we have the generalized result for the Bezoutian satisfying Hessenberg displacement structure $C_Q^T S_Q - S_Q C_Q = 0$ let us state the Schur–Euclid–Hessenberg algorithm to recover the coefficients of the remainder $c(x)$ of the polynomial division of $a(x)$ by $b(x)$ over basis $\{Q\} = \{Q_k\}_{k=0}^n$ where $\deg Q_k(x) = k$. In the meantime we will be providing triangular factorization of the Bezoutian ($S_Q = [\hat{s}_{ij}] = LDU$) over basis $\{Q\}$. The k-th row u_k of U and the k-th column l_k of L are given by $u_k = \frac{1}{\hat{s}_{11}^k} \left[0 \; \hat{s}_{1,\cdot}^{(k)} \right]$ and $l_k = u_k^T$ where "0" stands for a zero vector of appropriate length. The diagonal factor is given by $D = diag\,[\hat{s}_{11}^{(k)}]_{k=1}^n$.

The Schur–Euclid–Hessenberg Algorithm

Input: Coefficients of $a(x)$ and $b(x)$, say $a_0, a_1, \cdots, a_n$ and $b_0, b_1, \cdots, b_{n-1}$, and if the degree of $b(x)$ is $n - k < n - 1$ then list its coefficients as $b_0, b_1, \cdots, b_{n-k}, 0$. Here "0" means a zero vector of appropriate length up to $n - 1$.

Initialization: $\hat{s}^{(1)}_{1,\cdot} = \hat{s}_{1,\cdot}$, $\hat{s}^{(1)}_{\cdot,1} = \hat{s}^{T}_{1,\cdot}$, $C^{(1)}_{Q} = C_Q$ in (27)

Recursion: For $k = 1, \cdots, n-1$ compute

1. The k-th entry D, the k-th row of U, and k-th column of L by
   $d^{(k)} = \hat{s}^{(k)}_{11}$, $u^{(k)} = \frac{1}{d^{(k)}}\hat{s}^{(k)}_{1,\cdot}$, $l^{(k)} = [u^{(k)}]^T$
2. The second row and column of $S^{(k)}_{Q}$ by
   If $k = 1$
   $\hat{s}^{(k)}_{2,\cdot} = \hat{s}^{(k)}_{1,\cdot} \cdot \left[C^{(1)}_{Q} - \left(r_{n-1,\,n} - \frac{a_{n-1}}{a_n}\right) I\right]$, $\hat{s}^{(k)}_{\cdot,2} = [\hat{s}^{(k)}_{2,\cdot}]^T$
   else
   $\hat{s}^{(k)}_{2,\cdot} = \hat{s}^{(k)}_{1,\cdot} \cdot \left[C^{(k)}_{Q} - r_{n-k,\,n-k+1} I\right]$, $\hat{s}^{(k)}_{\cdot,2} = [\hat{s}^{(k)}_{2,\cdot}]^T$
3. The first row and column of $S^{(k+1)}_{Q}$ which is the Schur complement of $S^{(k)}_{Q}$ by
   $\hat{s}^{(k+1)}_{1,\cdot} = \hat{s}^{(k)'}_{2,\cdot} - \hat{s}^{(k)}_{21}\left(\frac{1}{\hat{s}^{(k)}_{11}}\hat{s}^{(k)}_{1,\cdot}\right)'$, $\hat{s}^{(k+1)}_{\cdot,1} = [\hat{s}^{(k+1)}_{1,\cdot}]^T$
   Here the prime means the first component is peeled off. (If deg $b(x) = n - k$ peel off the first k components for the first step).
4. Coefficients of the remainder $c(x)$ by
   $c^{(k+1)}_{\cdot,1} = \frac{1}{\hat{s}^{(k+1)}_{11}}\hat{s}^{(k+1)}_{\cdot,1}$
5. New Confederate matrix generated by
   $C^{(k+1)}_{Q} = C^{(k)''}_{Q} - (e_1)'\left(\frac{1}{\hat{s}^{(k)}_{11}}\hat{s}^{(k)'}_{1,\cdot}\right)$

Here double prime means peel off the top row and the first column of the matrix. (If deg $b(x) = n - k$ peel off the first k rows and columns of the matrix for the first step).

Output: Coefficients of the remainder and triangular factorization of the generalized Bezoutian

Proposition 1 *The arithmetic cost of computing the Schur–Euclid–Hessenberg algorithm for a generalized Bezoutian is $\mathcal{O}(M(n)n)$ where $M(n)$ is the cost of multiplying the confederate matrix C_Q by vectors ($k = 1, 2, \cdots, n$).*

Due to the upper Hessenberg structure of the confederate matrix C_Q in the above scenario $M(n) = n^2$. Thus to derive a fast Schur–Euclid–Hessenberg algorithm one has to analyze the confederate matrices based on quasiseparable polynomials or their subclasses as orthogonal and Szegö polynomials as defined in the next section.

5 A Fast Schur–Euclid Algorithm for Quasiseparable Polynomials

We have seen the Schur–Euclid–Hessenberg algorithm for generalized Bezoutian associated with the polynomials expanded over the basis $\{Q_k(x)\}_{k=0}^{n}$ where deg $Q_k(x) = k$ and its arithmetic complexity. The cost of Schur–Euclid–Hessenberg

algorithm is determined by the cost of the multiplication of the confederate matrix by vectors. Thus the arithmetic complexity of the algorithm can be reduced having sparse, banded, or structured confederate matrices. Hence in this section, we discuss the complexity of Schur–Euclid–Hessenberg algorithm for quasiseparable polynomials and therefore its sub classes [7]: orthogonal and Szegö polynomials while elaborating a fast Schur–Euclid–Hessenberg algorithm for quasiseparable polynomials.

5.1 Schur–Euclid–Hessenberg Algorithm for Quasiseparable Polynomials

In this section, we analyze the matrix decomposition for the confederate matrix over quasiseparable polynomials and use the decomposition to derive a fast Schur–Euclid–Hessenberg algorithm for a Bezoutian associated with the quasiseparable polynomials. The ultimate idea is to explore the confederate matrix over quasiseparable polynomials to reduce the cost of computing $M(n)$ in Proposition 1.

Let us start with the generator definition of $(H, 1)$-quasiseparable matrix which is equivalent to the rank Definition 2. We use quasiseparable generators to define a system of quasiseparable polynomials.

Definition 8 A matrix A is called $(H, 1)$-quasiseparable if **(i)** it is strongly upper Hessenberg, and **(ii)** it can be represented in the form

$$A = \begin{bmatrix} d_1 & & & \\ q_1 & d_2 & & g_i b_{ij}^{\times} h_j \\ 0 & q_2 & \ddots & \\ \vdots & \ddots & \ddots & \ddots \\ 0 & \cdots & 0 & q_{n-1} \quad d_n \end{bmatrix}$$

where $b_{ij}^{\times} = b_{i+1} b_{i+2} \cdots b_{j-1}$ for $j > i+1$ and $b_{ij}^{\times} = 1$ for $j = i+1$. The scalar elements $\{q_k, d_k, g_k, b_k, h_k\}$ are called the generators of the matrix A.

The results of [7, 37] allow one to observe a bijection between the set of strongly upper Hessenberg matrices $\mathscr{H}$ (say $A = [a_{ij}] \in \mathscr{H}$) and the set of polynomials system $\mathscr{P}$ (say $R = \{r_k(x)\} \in \mathscr{P}$ with $\deg r_k(x) = k$) via

$$f : \mathscr{H} \to \mathscr{P}, \text{where } r_k(x) = \frac{1}{a_{2,1} a_{3,2} \cdots a_{k,k-1}} det(xI - A)_{k \times k}. \tag{33}$$

The following lemma is given in [7, 8] and is a consequence of Definition 8 and [37].

Lemma 10 *Let A be an $(H, 1)$-quasiseparable matrix specified by its generators as in Definition 8. Then a system of polynomials $\{r_k(x)\}$ satisfies the recurrence relations*

$$r_k(x) = \frac{1}{q_k}\left[(x-d_k)r_{k-1}(x) - \sum_{j=0}^{k-2} g_{j+1}b^{\times}_{j+1,k}h_k\, r_j(x)\right] \tag{34}$$

if and only if $\{r_k(x)\}$ *is related to A via* (33).

With the help of the system of quasiseparable polynomials $\{r_k(x)\}_{k=0}^{n}$ satisfying k-term recurrence relations (34), we will define a confederate matrix for the polynomial

$$a(x) = a_0 r_0(x) + a_1 r_1(x) + \cdots + a_n r_n(x) \tag{35}$$

by

$$C_R(a) = \begin{bmatrix} d_1 & g_1h_2 & g_1b_2h_3 & \cdots & \cdots & g_1b^{\times}_{1,n}h_n - \frac{a_0}{a_n} \\ q_1 & d_2 & g_2h_3 & \cdots & \cdots & g_2b^{\times}_{2,n}h_n - \frac{a_1}{a_n} \\ 0 & q_2 & d_3 & \cdots & \cdots & g_3b^{\times}_{3,n}h_n - \frac{a_1}{a_n} \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & q_{n-2} & d_{n-1} & g_{n-1}h_n - \frac{a_{n-2}}{a_n} \\ 0 & \cdots & \cdots & 0 & q_{n-1} & d_n - \frac{a_{n-1}}{a_n}. \end{bmatrix} \tag{36}$$

The matrix (36) is an upper Hessenberg matrix so following Theorem 4 the Bezoutian associated with quasiseparable polynomials satisfies $C_R^T S_R - S_R C_R = 0$, where R is the system of quasiseparable polynomials satisfying recurrence relations (34), $C_R = \tilde{I}[C_R(a)]^T\tilde{I}$, and $a(x)$ is defined via (35). Though one can state a Schur–Euclid–Hessenberg algorithm for a Bezoutian associated with quasiseparable polynomials using the Schur–Euclid–Hessenberg algorithm in Sect. 4.2 it is not cheap because the structure of the confederate matrix C_R is not sparse. Thus to reduce the cost of the Schur–Euclid–Hessenberg algorithm for a Bezoutian associated with quasiseparable polynomials one has to explore the structure of the confederate matrix C_R via matrix decomposition.

Theorem 5 *Let* C_R *be the matrix specified by generators* $\{q_k, d_k, g_k, b_k, h_k\}$ *and coefficients* a_k *via*

$$C_R = \begin{bmatrix} d_n - \frac{a_{n-1}}{a_n} & g_{n-1}h_n - \frac{a_{n-2}}{a_n} & g_{n-2}b_{n-1}h_n - \frac{a_{n-3}}{a_n} & \cdots\cdots & g_1b^{\times}_{1,n}h_n - \frac{a_0}{a_n} \\ q_{n-1} & d_{n-1} & g_{n-2}h_{n-1} & \cdots\cdots & g_1b^{\times}_{1,n-1}h_{n-1} \\ 0 & q_{n-2} & d_{n-2} & \cdots\cdots & g_1b^{\times}_{1,n-2}h_{n-2} \\ \vdots & \ddots & \ddots & \ddots\ \ddots & \vdots \\ \vdots & \ddots & \ddots & q_2\ \ d_2 & g_1h_2 \\ 0 & \cdots & \cdots & 0\ \ q_1 & d_1 \end{bmatrix} \tag{37}$$

then the following decomposition holds:

$$C_R = \left[\tilde{\theta}_n\left(\cdots\left(\tilde{\theta}_3\left(\tilde{\theta}_2\tilde{\theta}_1 + \tilde{\Delta}_2\right) + \tilde{\Delta}_3\right)\cdots\right) + \tilde{\Delta}_n\right] + \frac{1}{a_n}\cdot\tilde{A}_1\tilde{A}_2\cdots\tilde{A}_n \tag{38}$$

where

$$\tilde{\theta}_1 = \begin{bmatrix} I_{n-2} & & \\ & d_2 & g_1 \\ & q_1 & d_1 \end{bmatrix}, \quad \tilde{\theta}_k = \begin{bmatrix} I_{n-k-1} & & & \\ & d_{k+1} & b_k & \\ & q_k & h_k & \\ & & & I_{k-1} \end{bmatrix}, \quad \tilde{\theta}_n = \begin{bmatrix} h_n & \\ & I_{n-1} \end{bmatrix}$$

$$\tilde{\Delta}_k = \begin{bmatrix} 0_{n-k-1} & & & \\ & 0 & g_k - d_k b_k & \\ & 0 & d_k - d_k h_k & \\ & & & 0_{k-1} \end{bmatrix}, \quad \tilde{\Delta}_n = \begin{bmatrix} d_n - d_n h_n & \\ & 0_{n-1} \end{bmatrix} \tag{39}$$

and

$$\tilde{A}_k = \begin{bmatrix} I_{k-1} & & & \\ & -a_{n-k} & 1 & \\ & 0 & 0 & \\ & & & I_{n-k-1} \end{bmatrix}, \quad \tilde{A}_n = \begin{bmatrix} I_{n-1} & \\ & -a_0. \end{bmatrix} \tag{40}$$

Proof Let us split the matrix C_R into $C_R = \tilde{H} + \frac{1}{a_n} C$ where

$$\tilde{H} = \begin{bmatrix} d_n & g_{n-1}h_n & g_{n-2}b_{n-1}h_n & \cdots\cdots & g_1 b^{\times}_{1,n} h_n \\ q_{n-1} & d_{n-1} & g_{n-2}h_{n-1} & \cdots\cdots & g_1 b^{\times}_{1,n-1} h_{n-1} \\ 0 & q_{n-2} & d_{n-2} & \cdots\cdots & g_1 b^{\times}_{1,n-2} h_{n-2} \\ \vdots & \ddots & \ddots & \ddots\ \ddots & \vdots \\ \vdots & \ddots & \ddots & q_2 \quad d_2 & g_1 h_2 \\ 0 & \cdots & \cdots & 0 \quad q_1 & d_1 \end{bmatrix}$$

and $C = \begin{bmatrix} -a_{n-1} & -a_{n-2} & \cdots & -a_1 & -a_0 \\ 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ . & . & . & . & . \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}$. One can easily show by matrix multiplication that the latter matrix admits the factorization $C = \tilde{A}_1 \tilde{A}_2 \cdots \tilde{A}_n$ with the given $\tilde{A}_k$ for $k = 1, 2, \cdots, n$. Thus we only have to prove the decomposition for $\tilde{H}$ in terms of $\tilde{\theta}_k$'s and $\tilde{\Delta}_k$'s.

Showing the decomposition for $\tilde{H} = \tilde{\theta}_n \left(\cdots \left(\tilde{\theta}_3 \left(\tilde{\theta}_2 \tilde{\theta}_1 + \tilde{\Delta}_2 \right) + \tilde{\Delta}_3 \right) \cdots \right) + \tilde{\Delta}_n$ is equivalent to showing that the matrix $\tilde{H}$ satisfies the iteration:

$$\tilde{H}_0 = I_n, \ \tilde{H}_k = \tilde{\theta}_k \tilde{H}_{k-1} + \tilde{\Delta}_k, \ k = 1, 2, \cdots, n, \ \tilde{H} = \tilde{H}_n. \tag{41}$$

Let us show by induction that for every $k = 3, 4, \cdots, n$:

$$\tilde{H}_{k-1}(n-k+1:n, n-k+1:n) = \tilde{H}(n-k+1:n, n-k+1:n) \tag{42}$$

The basis of induction (k=3) is trivial

$$\tilde{H}(n-2:n,n-2:n)=\begin{bmatrix} d_2 & g_1h_2 \\ q_1 & d_1 \end{bmatrix}=\tilde{H}_2(n-2:n,n-2:n).$$

Assume that (42) holds for all indices up to k. Consider the matrix $\tilde{H}_k(n-k+2:n,n-k+2:n)$:

$$\begin{bmatrix} d_{k+1} & b_k & \\ q_k & h_k & \\ & & I_{k-1} \end{bmatrix}\left[\begin{array}{c|ccc} 1 & 0 & \cdots & 0 \\ \hline 0 & & & \\ \vdots & & \tilde{H}_{k-1}(n-k+1:n,n-k+1:n) & \\ 0 & & & \end{array}\right]+\begin{bmatrix} 0 & g_k-d_kb_k & \\ 0 & d_k-d_kh_k & \\ & & 0_{k-1}. \end{bmatrix} \tag{43}$$

The first row of the matrix $\tilde{H}_{k-1}(n-k+1:n,n-k+1:n)$ equals the first row of the matrix $\tilde{H}(n-k+1:n,n-k+1:n)$. Therefore, performing the matrix product in (43) we get:

$$\left[\begin{array}{c|c|c} d_{k+1} & d_kb_k+(g_k-d_kb_k) & \tilde{H}_{k-1}(n-k-2,n-k:n)b_k \\ \hline q_k & d_kh_k+(d_k-d_kh_k) & \tilde{H}_{k-1}(n-k-1,n-k:n)h_k \\ \hline 0 & & \\ \vdots & \tilde{H}_{k-1}(n-k:n,n-k-1) & \tilde{H}_{k-1}(n-k:n,n-k:n) \\ 0 & & \end{array}\right]$$

which is equal to $\tilde{H}(n-k+2:n,n-k+2:n)$. By induction we get $\tilde{H}_{n-1}(1:n,1:n)=\tilde{H}(1:n,1:n)$. Substituting this into recursion (41) we get

$$\tilde{H}_n=\begin{bmatrix} h_n & \\ & I_{n-1} \end{bmatrix}\tilde{H}_{n-1}+\begin{bmatrix} d_n-d_nh_n & \\ & 0_{n-1} \end{bmatrix}=\tilde{H}.$$

Now we have the decomposition of the confederate matrix C_R (38) over quasiseparable polynomials and Hessenberg-1-quasiseparable displacement structure of the Bezoutian $C_R^TS_R-S_RC_R=0$. Thus the following Schur–Euclid–Hessenberg algorithm for a Bezoutian associated with quasiseparable polynomials can be used to recover coefficients of the remainder $c(x)$ of the polynomial division $a(x)$ by $b(x)$ over the basis $\{R\}=\{r_k\}_{k=0}^n$ where deg $r_k(x)=k$ and $\{R\}$ satisfies the recurrence relations (34), and also to obtain the triangular factorization of Bezoutian over the system of quasiseparable polynomials $\{R\}$.

The Schur–Euclid–Hessenberg Algorithm for Bezoutian Over Quasiseparabale polynomials

Input: Generators $\{q_k,d_k,g_k,b_k,h_k\}$. Coefficients of $a(x)$ and $b(x)$, say $a_0,a_1,\cdots,a_n$ and $b_0,b_1,\cdots,b_{n-1}$, and if the degree of $b(x)$ is $n-k<n-1$ then list its coefficients as $b_0,b_1,\cdots,b_{n-k},0$ here "0" means a zero vector of appropriate length up to $n-1$.

Initialization: Set $\tilde{\theta}_k$ and $\tilde{\Delta}_k$ in terms of generators $\{q_k, d_k, g_k, b_k, h_k\}$, and $\tilde{A}_k$ in terms of a_k and $C_R^{(1)} = C_R$ via (38). Set $\hat{s}_{.,1}^{(1)} = \hat{s}_{.,1}$.

Recursion: For $k = 1, \cdots, n-1$ compute

1. The k-th entry D, the k-th row of U, and k-th column of L by
   $d^{(k)} = \hat{s}_{11}^{(k)}, \quad u^{(k)} = \frac{1}{d^{(k)}} \hat{s}_{1,\cdot}^{(k)}, \quad l^{(k)} = [u^{(k)}]^T$
2. The second row and column of $S_R^{(k)}$ by
   If $k = 1$
   $\hat{s}_{2,\cdot}^{(k)} = \frac{1}{q_{n-1}} \hat{s}_{1,\cdot}^{(k)} \left[C_R^{(1)} - \left(d_n - \frac{a_{n-1}}{a_n} \right) I \right], \quad \hat{s}_{\cdot,2}^{(k)} = [\hat{s}_{2,\cdot}^{(k)}]^T$
   else
   $\hat{s}_{2,\cdot}^{(k)} = \frac{1}{q_{n-k}} \hat{s}_{1,\cdot}^{(k)} \left[C_R^{(k)} - d_{n-k+1} I \right], \quad \hat{s}_{\cdot,2}^{(k)} = [\hat{s}_{2,\cdot}^{(k)}]^T$
3. The first row and column of $S_R^{(k+1)}$ which is the Schur complement of $S_R^{(k)}$ by
   $\hat{s}_{1,\cdot}^{(k+1)} = \hat{s}_{2,\cdot}^{(k)'} - \hat{s}_{21}^{(k)} \left(\frac{1}{\hat{s}_{11}^{(k)}} \hat{s}_{1,\cdot}^{(k)} \right)', \quad \hat{s}_{\cdot,1}^{(k+1)} = [\hat{s}_{1,\cdot}^{(k+1)}]^T$
   Here the prime means the first component is peeled off. (If deg $b(x) = n - k$, then peel off the first k components for the first step).
4. Coefficients of the remainder $c(x)$ by
   $c_{\cdot,1}^{(k+1)} = \frac{1}{\hat{s}_{11}^{(k+1)}} \hat{s}_{\cdot,1}^{(k+1)}$
5. Confederate matrix after peeling off row(s) and column(s) by
   $\tilde{C}_R^{(k)} = \tilde{\theta}_{n-k}'' \left(\cdots \left(\tilde{\theta}_3'' \left(\tilde{\theta}_2'' \tilde{\theta}_1'' + \tilde{\Delta}_2'' \right) + \tilde{\Delta}_3'' \right) \cdots \right) + \tilde{\Delta}_{n-k}''$
   Here the double prime means peel off the top row and the first column of matrices. (If deg $b(x) = n - k$, then peel off the first k rows and columns of the matrix for the first step).
6. New confederate matrix generated by
   $C_R^{(k+1)} = \tilde{C}_R^{(k)} - q_{n-k} (e_1)' \left(\frac{1}{\hat{s}_{11}^{(k)}} \hat{s}_{1,\cdot}^{(k)'} \right)$

Output: Coefficients of the remainder and triangular factorization of the Bezoutian associated with quasiseparable polynomials.

As we have seen in Proposition 1, the cost of the Schur–Euclid–Hessenberg algorithm is dominated by $M(n)$ which is the cost of multiplication of a confederate matrix by vectors and this occurs in step 2 of the algorithm. Also note that for the multiplication of $C_R^{(k)}$ by vectors, i.e., for $k = 1$ we have to multiply n factors of $\tilde{\theta}_k$, $n-1$ factors of $\tilde{\Delta}_k$, and n factors of $\tilde{A}_k$ together with the scaling factor $\frac{1}{a_n}$ (note that we do not have to scale for the monic polynomial case) by a vector and for $k = 2, 3, \cdots, n-1$ we have to multiply at most $n-k$ factors of $\tilde{\theta}$ and $n-k-1$ factors of $\tilde{\Delta}$ by a vector. Thus the most expensive step in the recursion is when $k = 1$. Now for $k = 1$ in step 2 of the Schur–Euclid–Hessenberg algorithm in the quasiseparable case, we have at most 4 multiplications and 2 additions corresponding to multiplication of $\tilde{\theta}_k$, at most 2 multiplications corresponding to multiplication of $\tilde{\Delta}_k$, and at most 1 multiplication and 1 addition corresponding to multiplication of $\tilde{A}_k$ by the first row of the Bezoutian. Thus the arithmetic cost of computing $C_R^{(1)}$ by a

vector is at most $11n - 2$ operations as opposed to Hessenberg structured matrix C_R (37) by a vector, which is $n^2 - 2$ operations. Hence $\forall n > 11$, one can design a fast Schur–Euclid–Hessenberg algorithm for quasiseparable polynomials.

Proposition 2 *The arithmetic cost of computing the Schur–Euclid–Hessenberg algorithm for the Bezoutian associated with the quasiseparable polynomials satisfying recurrence relations* (34) *is* $\mathcal{O}(n^2)$.

5.2 Cost of Schur–Euclid–Hessenberg Algorithm for Orthogonal Polynomials

In this section, we observe the cost of computing the Schur–Euclid–Hessenberg Algorithm for a Bezoutian associated with the orthogonal polynomials. The main idea here is to explore the confederate matrix with respect to the orthogonal polynomial system to reduce the cost of computing $M(n)$ in Proposition 1.

It is well known [16] that systems of polynomials $R = \{r_k(x)\}_{k=0}^n$ orthogonal with respect to an inner product of the form

$$< p(x), q(x) >= \int_a^b p(x)q(x)w^2(x)dx$$

satisfy a three-term recurrence relation of the form

$$r_k(x) = \frac{1}{q_k}(x - d_k)r_{k-1}(x) - \frac{g_{k-1}}{q_k} \cdot r_{k-2}(x), \quad q_k \neq 0. \tag{44}$$

Define for the polynomial

$$a(x) = a_0 r_0(x) + a_1 r_1(x) + \cdots + a_{n-1} r_{n-1}(x) + a_n r_n(x) \tag{45}$$

its confederate matrix, given by

$$C(a) = \begin{bmatrix} d_1 & g_1 & 0 & \cdots & 0 & -\frac{a_0}{a_n} \\ q_1 & d_2 & g_2 & \ddots & \vdots & -\frac{a_1}{a_n} \\ 0 & q_2 & d_3 & \ddots & 0 & -\frac{a_2}{a_n} \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & q_{n-2} & d_{n-1} & g_{n-1} - \frac{a_{n-2}}{a_n} \\ 0 & 0 & \cdots & 0 & q_{n-1} & d_n - \frac{a_{n-1}}{a_n} \end{bmatrix} \tag{46}$$

which has been called a comrade matrix in [4].

The matrix (46) is an upper Hessenberg matrix so by Theorem 4 the Bezoutian associated with orthogonal polynomials satisfies $C_R^T S_R - S_R C_R = 0$, where R is the system of orthogonal polynomials satisfying recurrence relations (44), $C_R = \tilde{I}C(a)^T\tilde{I}$, and $a(x)$ is defined via (45). Moreover orthogonal polynomials are a sub class of quasiseparable polynomials [7] so to express Schur–Euclid–Hessenberg algorithm for a Bezoutian associated with the orthogonal polynomials one has to revise the Schur–Euclid–Hessenberg Algorithm in the quasiseparable case in Sect. 5.1. To do so one has to consider the Bezoutian associated with orthogonal polynomials and initialize the algorithm with the comrade matrix $C_R = \tilde{I}C(a)^T\tilde{I}$ having generators $\{q_k, d_k, g_k\}$ with the coefficients a_k. Due to the sparse structure of the comrade matrix, we could ignore step 5 of the Schur–Euclid–Hessenberg Algorithm in the quasiseparable case and revise the first matrix in the RHS of step 6 to be the comrade matrix $C_R = \tilde{I}C(a)^T\tilde{I}$.

The matrix C_R is almost tridiagonal. Thus the cost of multiplication of C_R by vectors is only $M(n) = \mathcal{O}(n)$ operations. Hence one can design a fast Schur–Euclid–Hessenberg algorithm for a Bezoutian associated with orthogonal polynomials having complexity $\mathcal{O}(n^2)$ operations.

Proposition 3 *The arithmetic cost of computing the Schur–Euclid–Hessenberg algorithm for the Bezoutian associated with the orthogonal polynomials satisfying* (44) *is* $\mathcal{O}(n^2)$.

5.3 Cost of Schur–Euclid–Hessenberg Algorithm for Szegö Polynomials

In this section, we observe the cost of computing the Schur–Euclid–Hessenberg Algorithm for a Bezoutian associated with the Szegö polynomials. The main idea here is to explore the confederate matrix with respect to the Szegö polynomial system to reduce the cost of computing $M(n)$ in Proposition 1.

Szegö polynomials $S = \{\phi_k^{\sharp}(x)\}_{k=0}^n$ or polynomials orthonormal on the unit circle with respect to an inner product of the form

$$< p(x), q(x) >= \frac{1}{2\pi}\int_{-\pi}^{\pi} p(e^{i\theta})[q(e^{i\theta})]^* w^2(\theta)d\theta,$$

for any such inner product, it is known [22] that there exist a set of reflection coefficients $\{\rho_k\}$ satisfying

$$\rho_0 = -1, \quad |\rho_k| < 1, \quad k = 1, 2, \cdots, n-1, \quad |\rho_n| \leq 1,$$

and complementary parameters $\{\mu_k\}$ defined by the reflection coefficients via

$$\mu_k = \begin{cases} \sqrt{1-|\rho_k|^2}, & |\rho_k| < 1 \\ 1, & |\rho_k| = 1 \end{cases}$$

such that the corresponding Szegö polynomials satisfying the two-term recurrence relations

$$\begin{bmatrix} \phi_k(x) \\ \phi_k^\sharp(x) \end{bmatrix} = \frac{1}{\mu_0}\begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad \begin{bmatrix} \phi_k(x) \\ \phi_k^\sharp(x) \end{bmatrix} = \frac{1}{\mu_k}\begin{bmatrix} 1 & -\rho_k^\star \\ -\rho_k & 1 \end{bmatrix}\begin{bmatrix} \phi_{k-1}(x) \\ x\,\phi_{k-1}^\sharp(x) \end{bmatrix} \tag{47}$$

where $\{\phi_k(x)\}$ is a system of auxiliary polynomials. Define for the polynomial

$$a(x) = a_0\phi_0^\sharp(x) + a_1\phi_1^\sharp(x) + \cdots + a_{n-1}\phi_{n-1}^\sharp(x) + a_n\phi_n^\sharp(x) \tag{48}$$

its confederate matrix is given by

$$C_S(a) = \begin{bmatrix} -\rho_0^*\rho_1 & -\rho_0^*\mu_1\rho_2 & -\rho_0^*\mu_1\mu_2\rho_3 & \cdots & -\rho_0^*\mu_1\mu_2\cdots\mu_{n-1}\rho_n - \frac{a_0}{a_n} \\ \mu_1 & -\rho_1^*\rho_2 & -\rho_1^*\mu_2\rho_3 & \cdots & -\rho_1^*\mu_2\mu_3\cdots\mu_{n-1}\rho_n - \frac{a_1}{a_n} \\ 0 & \mu_1 & -\rho_2^*\rho_3 & \cdots & -\rho_2^*\mu_3\mu_4\cdots\mu_{n-1}\rho_n - \frac{a_1}{a_n} \\ \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \mu_{n-1} & -\rho_{n-1}^*\rho_n - \frac{a_{n-1}}{a_n}. \end{bmatrix} \tag{49}$$

The matrix (49) is an upper Hessenberg matrix so following Theorem 4 the Bezoutian associated with Szegö polynomials satisfies $C_S^T S_S - S_S C_S = 0$ where S is the system of Szegö polynomials satisfying recurrence relations (47), $C_S = \tilde{I}[C_S(a)]^T\tilde{I}$, and $a(x)$ is defined via (48). The matrix C_S is not sparse like the orthogonal polynomial case so to reduce the cost of multiplication of C_S by vectors one has to use the nested factorization of C_S.

Lemma 11 *Let C_S be the matrix specified by generators $\{\rho_k^*, \rho_k, \mu_k\}$ and coefficients a_k via*

$$C_S = \tilde{I}[C_S(a)]^T\tilde{I} \tag{50}$$

then the following decomposition holds:

$$C_S = \tilde{\Gamma}_0\tilde{\Gamma}_1\tilde{\Gamma}_2\cdots\tilde{\Gamma}_n + \frac{1}{a_n}\cdot\tilde{A}_1\tilde{A}_2\cdots\tilde{A}_n \tag{51}$$

where

$$\tilde{\Gamma}_0 = \begin{bmatrix} -\rho_n & \\ & I_{n-1} \end{bmatrix}, \quad \tilde{\Gamma}_k = \begin{bmatrix} I_{k-1} & & & \\ & \rho_{n-k}^* & \mu_{n-k} & \\ & \mu_{n-k} & -\rho_{n-k} & \\ & & & I_{n-k-1} \end{bmatrix}, \quad \tilde{\Gamma}_n = \begin{bmatrix} I_{n-1} & \\ & \rho_0^* \end{bmatrix} \tag{52}$$

and

$$\tilde{A}_k = \begin{bmatrix} I_{k-1} & & & \\ & -a_{n-k} & 1 & \\ & 0 & 0 & \\ & & & I_{n-k-1} \end{bmatrix}, \quad \tilde{A}_n = \begin{bmatrix} I_{n-1} & \\ & -a_0. \end{bmatrix}$$

Proof The matrix C_S can be split into $C_S = \tilde{U} + \frac{1}{a_n} C$ where

$$\tilde{U} = \begin{bmatrix} -\rho_{n-1}^* \rho_n & -\rho_{n-2}^* \mu_{n-1} \rho_n & -\rho_{n-3}^\star \mu_{n-2} \mu_{n-1} \rho_n & \cdots & -\rho_0^* \mu_1 \mu_2 \cdots \mu_{n-1} \rho_n \\ \mu_{n-1} & -\rho_{n-2}^\star \rho_{n-1} & -\rho_{n-3}^\star \mu_{n-2} \rho_{n-1} & \cdots & -\rho_0^* \mu_1 \mu_2 \cdots \mu_{n-2} \rho_{n-1} \\ 0 & \mu_{n-2} & -\rho_{n-3}^\star \rho_{n-2} & \cdots & -\rho_0^* \mu_1 \mu_2 \cdots \mu_{n-3} \rho_{n-2} \\ \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \mu_1 & -\rho_0^\star \rho_1 \end{bmatrix}$$

and $C = \begin{bmatrix} -a_{n-1} & -a_{n-2} & \cdots & -a_1 & -a_0 \\ 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ . & . & . & . & . \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}$. Let U be the unitary Hessenberg matrix [9] corresponding to Szegö polynomials $\{\phi_k^\sharp(x)\}$ satisfying 2-term recurrence relations (47), then we can see that

$$\tilde{U} = \tilde{I} U^T \tilde{I}$$

It is well known that unitary Hessenberg matrix U can be written as the product $U = \Gamma_0 \Gamma_1 \Gamma_2 \cdots \Gamma_n$ where

$$\Gamma_0 = \begin{bmatrix} \rho_0^* & \\ & I_{n-1} \end{bmatrix}, \quad \Gamma_k = \begin{bmatrix} I_{k-1} & & & \\ & -\rho_k & \mu_k & \\ & \mu_k & \rho_k^* & \\ & & & I_{n-k-1} \end{bmatrix}, \quad \Gamma_n = \begin{bmatrix} I_{n-1} & \\ & -\rho_n \end{bmatrix}.$$

Thus $\tilde{U}$ has the factorization

$$\tilde{\Gamma}_0 \tilde{\Gamma}_1 \tilde{\Gamma}_2 \cdots \tilde{\Gamma}_n$$

where $\tilde{\Gamma}_k = \tilde{I} \Gamma_k^T \tilde{I}$ for $k = 0, 1, \cdots, n$.

One can see clearly by the matrix multiplication that the matrix C admits the factorization

$$C = \tilde{A}_1 \tilde{A}_2 \cdots \tilde{A}_n$$

where $\tilde{A}_k = \begin{bmatrix} I_{k-1} & & & \\ & -a_{n-k} & 1 & \\ & 0 & 0 & \\ & & & I_{n-k-1} \end{bmatrix}, \quad \tilde{A}_n = \begin{bmatrix} I_{n-1} & \\ & -a_0 \end{bmatrix}$, hence the result.

Szegö polynomials are a sub class of quasiseparable polynomials [7] so to express the Schur–Euclid–Hessenberg algorithm for the Bezoutian associated with the Szegö polynomials one has to revise the Schur–Euclid–Hessenberg Algorithm in the quasiseparable case, in Sect. 5.1. To do so one has to consider the Bezoutian associated with Szegö polynomials and initialize the algorithm with the confederate matrix (51) having generators $\{\rho_k^*, \rho_k, \mu_k\}$ with the coefficients a_k. But to reduce the complexity $M(n)$ one has to revise step 5 of the Schur–Euclid–Hessenberg Algorithm for the quasiseparable case with the decomposition $\tilde{C}_S^{(k)} = \tilde{\Gamma}_k'' \tilde{\Gamma}_{k+1}'' \cdots \tilde{\Gamma}_n''$ where $\tilde{\Gamma}_k$'s are given in (52) and the double prime means peel off the top k column(s) and row(s) if deg $b(x) = n - k$ where deg $a(x) = n$ while revising the first matrix in the RHS of step 6 to be the confederate matrix $\tilde{C}_S^{(k)}$ in step 5.

Due to the decomposition of C_S, in step 2 of the Schur–Euclid–Hessenberg algorithm for Szegö polynomials, we have at most 4 multiplications and 2 additions corresponding to multiplication of $\tilde{\Gamma}_k$ by the first row of the Bezoutian, and at most 1 multiplication and 1 addition corresponding to multiplication of $\tilde{A}_k$ by the first row of the Bezoutian. Thus the total cost of multiplication of $n + 1$ factors of $\tilde{\Gamma}_k$ which is $\tilde{U}$ by the vector is $6(n + 1)$ operations, and n factors of $\tilde{A}_k$ which is C by the vector costs $2n$ operations. Together with the multiplication of C by quantity $\frac{1}{a_n}$ gives the overall cost of multiplication of C_S by vectors as $M(n) = \mathcal{O}(n)$ complexity. Hence one can design a fast Schur–Euclid–Hessenberg algorithm for Szegö polynomials.

Proposition 4 *The arithmetic cost of computing the Schur–Euclid–Hessenberg algorithm for a Bezoutian associated with the Szegö polynomials satisfying recurrence relations* (47) *is* $\mathcal{O}(n^2)$.

6 Conclusion

In this paper, we have derived a Schur–Euclid–Hessenberg algorithm to compute the triangular factorization of a generalized Bezoutian. In this case, it is associated with the system of polynomials $\{Q\} = \{Q_k(x)\}_{k=0}^n$, where deg $Q_k(x) = k$ and satisfies k-term recurrence relations while recovering the coefficients of the remainder of the polynomial division over basis $\{Q\}$. Once the generalization results were established, we explore a fast Schur–Euclid–Hessenberg algorithm for quasiseparable polynomials. This algorithm generalizes the result for fast Schur–Euclid–Hessenberg algorithm for orthogonal polynomials and Szegö polynomials. To derive the fast algorithm we exploit the decomposition of confederate matrices over quasiseparable and Szegö polynomials and use the sparse comrade matrix for orthogonal polynomials. The presented Schur–Euclid–Hessenberg algorithm enables us to compute a fast triangular factorization of the Bezoutian associated with quasiseparable, Szegö, and orthogonal polynomials, and to recover coefficients of the remainder in polynomial division over quasiseparable, Szegö, and orthogonal basis with complexity $\mathcal{O}(n^2)$ operations.

References

1. Allen, B.M., Rosenthal, J.: A matrix Euclidean algorithm induced by state space realization. Linear Algebra Appl. **288**, 105–121 (1999)
2. Barnett, S.: Greatest common divisor of two polynomials. Linear Algebra Appl. **3**(1), 7–9 (1970)
3. Barnett, S.: A note on the Bezoutian matrix. SIAM J. Appl. Math. **22**(1), 84–86 (1972)
4. Barnett, S.: A companion matrix analogue for orthogonal polynomials. Linear Algebra Appl. **12**(3), 197–202 (1975)
5. Beckermann, B., Labahn, G.: When are two numerical polynomials relatively prime? J. Symb. Comput. **26**(6), 677–689 (1998)
6. Beckermann, B., Labahn, G.: A fast and numerically stable Euclidean-like algorithm for detecting relatively prime numerical polynomials. J. Symb. Comput. **26**(6), 691–714 (1998)
7. Bella, T., Eidelman, Y., Gohberg, I., Olshevsky V.: Classifications of three-term and two-term recurrence relations via subclasses of quasiseparable matrices. SIAM J. Matrix Anal.
8. Bella, T., Eidelman, Y., Gohberg, I., Olshevsky, V., Tyrtyshnikov, E.: Fast inversion of polynomial-Vandermonde matrices for polynomial systems related to order one quasiseparable matrices. In: Kaashoek, M.A., Rodman, L., Woerdeman, H.J. (eds.) Advances in Structured Operator Theory and Related Areas, Operator Theory: Advances and Applications, vol. 237, pp. 79–106. Springer, Basel (2013)
9. Bella, T., Olshevsky, V., Zhlobich, P.: Classifications of recurrence relations via subclasses of (H, k)-quasiseparable matrices. In: Van Dooren, P., Bhattacharyya, S.P., Chan, R.H., Olshevsky, V., Routray, A. (eds.) Numerical Linear Algebra in Signals, Systems and Control. Lecture Notes in Electrical Engineering, vol. 80, pp. 23–54. Springer, Netherlands (2011)
10. Bini, D.A., Boito, P.: Structured matrix-based methods for polynomial ε-GCD: analysis and comparisons. In: D'Andrea, C., Mourrain, B. (eds.) Proceedings of the 2007 International Symposium on Symbolic and Algebraic Computation, Waterloo, Canada, July 29–August 01 2007 (ISAAC '07), pp. 9–16. ACM, New York (2007)
11. Bini, D.A., Gemignani, L.: Fast parallel computation of the polynomial remainder sequence via Bezout and Hankel matrices. SIAM J. Comput. **24**(1), 63–77 (1995)
12. Bini, D.A., Gemignani, L.: Bernstein–Bezoutian matrices. Theor. Comput. Sci. **315**(2–3), 319–333 (2004)
13. Bitmead, R.R., Kung, S.Y., Anderson, B.D., Kailath, T.: Greatest common divisor via generalized Sylvester and Bezout matrices. IEEE Trans. Autom. Control **23**(6), 1043–1047 (1978)
14. Brown, W.S.: On Euclid's algorithm and the computation of polynomial greatest common divisors. J. Assoc. Comput. Mach. **18**(4), 478–504 (1971)
15. Brown, W.S., Traub, J.F.: On Euclid's algorithm and the theory of subresultants. J. Assoc. Comput. Mach. **18**(4), 505–514 (1971)
16. Calvetti, D., Reichel, L.: Fast inversion of vandermondelike matrices involving orthogonal polynomials. BIT Numer. Math. **33**(3), 473–484 (1993)
17. Delosme, J.M., Morf, M.: Mixed and minimal representations for Toeplitz and related systems, In: Proceedings 14th Asilomar Conference on Circuits, Systems, and Computers, Monterey, California (1980)
18. Dym, H.: Structured matrices, reproducing kernels and interpolation. In: Olshevsky, V. (ed.) Structured Matrices in Mathematics, Computer Science and Engineering I: Contemporary Mathematics, vol. 280, p. 329. American Mathematical Society, Providence (2001)
19. Genin, Y.V.: Euclid algorithm, orthogonal polynomials, and generalized Routh–Hurwitz algorithm. Linear Algebra Appl. **246**, 131–158 (1996)
20. Gohberg, I., Olshevsky, V.: Fast inversion of Chebyshev–Vandermonde matrices. Numer. Math. **67**(1), 71–92 (1994)
21. Gohberg, T., Kailath, T., Olshevsky, V.: Fast Gaussian elimination with partial pivoting for matrices with displacement structure. Math. Comput. **64**(212), 1557–1576 (1995)
22. Grenander, U., Szegö, G.: Toeplitz Forms and Applications. University of California Press, Berkeley (1958)

23. Heinig, G.: Matrix representations of Bezoutians. Linear Algebra Appl. **223–224**, 337–354 (1995)
24. Heinig, G., Olshevsky, V.: The Schur algorithm for matrices with Hessenberg displacement structure. In: Olshevsky, V. (ed.) Structured Matrices in Mathematics, Computer Science, and Engineering II: Contemporary Mathematics, vol. 281, pp. 3–16. American Mathematical Society, Providence (2001)
25. Heinig, G., Rost, K.: Algebraic Methods for Toeplitz-Like Matrices and Operators. Operator Theory: Advances and Applications, vol. 13. Birkhäuser, Basel (1984)
26. Heinig, G., Rost, K.: On the inverses of Toeplitz-plus-Hankel matrices. Linear Algebra Appl. **106**, 39–52 (1988)
27. Heinig, G., Rost, K.: Matrix representations of Toeplitz-plus-Hankel matrix inverses. Linear Algebra Appl. **113**, 65–78 (1989)
28. Heinig, G., Rost, K.: Split algorithm and ZW-factorization for Toeplitz and Toeplitz-plus-Hankel matrices. In: Proceedings of the Fifteenth International Symposium on Mathematical Theory of Networks and Systems, Notre Dame, August 12–16 (2002)
29. Heinig, G., Rost, K.: New fast algorithms for Toeplitz-plus-Hankel matrices. SIAM J. Matrix Anal. Appl. **25**(3), 842857 (2004)
30. Heinig, G., Rost, K.: Fast algorithms for Toeplitz and Hankel matrices. Linear Algebra Appl. **435**(1), 159 (2011)
31. Kailath, T., Kung, S.Y., Morf, M.: Displacement ranks of matrices and linear equations. J. Math. Anal. Appl. **68**(2), 395407 (1979)
32. Kailath, T., Olshevsky, V.: Displacement-structure approach to polynomial Vandermonde and related matrices. Linear Algebra Appl. **261**, 49–90 (1997)
33. Kailath, T.: Displacement structure and array algorithms. In: Kailath, T., Sayed, A.H. (eds.) Fast Reliable Algorithms for Matrices with Structure. SIAM, Philadelphia (1999)
34. Kailath, T., Sayed, A.H.: Displacement structure: theory and applications. SIAM Rev. **37**(3), 297–386 (1995)
35. Kaltofen, E., May, J., Yang, Z., Zhi, L.: Approximate factorization of multivariate polynomials using singular value decomposition. J. Symb. Comput. **43**(5), 359–376 (2008)
36. Lancaster, P., Tismenetsky, M.: The Theory of Matrices with Applications, 2nd edn. Academic Press INC., Orlando (1985)
37. Maroulas, J., Barnett, S.: Polynomials with respect to a general basis. I. Theory. J. Math. Anal. Appl. **72**, 177–194 (1979)
38. Morf, M.: Fast Algorithms for Multivariable Systems. Ph. D. Thesis, Stanford University (1974)
39. Noda, M.T., Sasaki, T.: Approximate GCD and its application to ill-conditioned algebraic equations. J. Comput. Appl. Math. **38**, 335–351 (1991)
40. Olshevsky, V., Stewart, M.: Stable factorization of Hankel and Hankel-like matrices. Numer. Linear Algebra **8**(6–7), 401–434 (2001)
41. Pan, V.Y.: Computation of approximate polynomial GCDs and an extension. Inf. Comput. **167**(2), 71–85 (2001)
42. Pan, V.Y., Tsigaridas, E.: Nearly optimal computations with structured matrices. In: Watt, S.M., Verschelde, J., Zhi, L. (eds.) Proceedings of the International Conference on Symbolic Numeric Computation, China, July 2014, pp. 21–30. ACM Digital Library, New York (2014)
43. Rost, K.: Generalized companion matrices and matrix representations for generalized Bezoutians. Linear Algebra Appl. **193**(1), 151–172 (1993)
44. Wimmer, H.K.: On the history of the Bezoutian and the resultant matrix. Linear Algebra Appl. **128**, 27–34 (1990)
45. Yang, Z.H.: Polynomial Bezoutian matrix with respect to a general basis. Linear Algebra Appl. **331**, 165–179 (2001)
46. Zarowski, C.J.: A Schur Algorithm for Strongly Regular Toeplitz-plus-Hankel Matrices. In: Proceedings of the 33rd Midwest Symposium on Circuits and Systems, Alta, Aug, 1990. IEEE Xplore Digital Library, vol. 1, pp. 556–559. IEEE, New York (1990)