

A FAST BJÖRCK–PEREYRA-TYPE ALGORITHM FOR SOLVING HESSENBERG-QUASISEPARABLE-VANDERMONDE SYSTEMS*

T. BELLA[†], Y. EIDELMAN[‡], I. GOHBERG[‡], I. KOLTRACHT[§], AND V. OLSHEVSKY[¶]

Israel Koltracht passed away on Feb. 17, 2008, and the surviving authors dedicate this paper to his memory.

Abstract. A fast $\mathcal{O}(n^2)$ algorithm is derived for solving linear systems where the coefficient matrix is a polynomial-Vandermonde matrix $V_R(x) = [r_{j-1}(x_i)]$ with polynomials $\{r_k(x)\}$ defined by a Hessenberg matrix with quasiseparable structure. The result generalizes the well-known Björck–Pereyra algorithm for classical Vandermonde systems involving monomials. It also generalizes the algorithms of Reichel–Opfer for $V_R(x)$ involving Chebyshev polynomials of Higham for $V_R(x)$ involving real orthogonal polynomials, and a recent algorithm of the authors for $V_R(x)$ involving Szegő polynomials. The new algorithm applies to a fairly general new class of (H, k) -quasiseparable polynomials (Hessenberg, order k quasiseparable) that includes (along with the above mentioned classes of real orthogonal and Szegő polynomials) several other important classes of polynomials, e.g., defined by banded Hessenberg matrices. Numerical experiments are presented that coincide with previous experiences with Björck–Pereyra-type algorithms giving better forward error than Gaussian elimination, and this accuracy is consistent with the so-called Chan–Foulser conditioning of the system.

Key words. Vandermonde matrices, Björck–Pereyra algorithm, quasiseparable matrices

AMS subject classifications. 65, 15

DOI. 10.1137/060676635

1. Introduction. The fact that the n^2 entries of Vandermonde matrices $V(x) = [x_i^{j-1}]$ are determined by only n parameters $\{x_k\}$ allows the design of fast algorithms for solving problems associated with those matrices. Specifically, an algorithm due to Björck and Pereyra [7] can solve the Vandermonde linear system $V(x)a = f$ in $\mathcal{O}(n^2)$ operations. This is as opposed to the $\mathcal{O}(n^3)$ operations required by the standard, structure-ignoring approach of Gaussian elimination. We start with a brief survey of their results.

1.1. The classical Björck–Pereyra algorithm. In [7], Björck and Pereyra derive a representation of the inverse $V(x)^{-1}$ of an $n \times n$ Vandermonde matrix as the product of bidiagonal matrices, that is,

$$(1.1) \quad V(x)^{-1} = U_1 \cdots U_{n-1} \cdot \tilde{L}_{n-1} \cdots \tilde{L}_1$$

and used this result to solve the linear system $V(x)a = f$ by computing the solution vector

$$a = U_1 \cdots U_{n-1} \cdot \tilde{L}_{n-1} \cdots \tilde{L}_1 f,$$

*Received by the editors December 4, 2006; accepted for publication (in revised form) by N. J. Higham February 11, 2009; published electronically DATE.

<http://www.siam.org/journals/simax/x-x/67663.html>

[†]Department of Mathematics, University of Rhode Island, Kingston, RI 02881-0816 (tombella@math.uri.edu).

[‡]School of Mathematical Sciences, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Ramat-Aviv 69978, Israel (eideyu@post.tau.ac.il, gohberg@post.tau.ac.il).

[§]This author is deceased. Former address: Department of Mathematics, University of Connecticut, Storrs, CT 06269-3009.

[¶]Department of Mathematics, University of Connecticut, Storrs, CT 06269-3009 (olshevsky@math.uconn.edu).

which solves the linear system in $\frac{5}{2}n^2$ operations. This is an order of magnitude improvement over Gaussian elimination, which is well known to require $\mathcal{O}(n^3)$ operations in general. This favorable complexity results from the fact that the matrices U_k and L_k are sparse. More specifically, the factors U_k and L_k are given by

$$(1.2) \quad U_k = \left[\begin{array}{c|ccc} I_{k-1} & & & \\ \hline & 1 & -x_k & \\ & & 1 & \ddots \\ & & & \ddots & -x_k \\ & & & & 1 \end{array} \right],$$

$$(1.3) \quad \tilde{L}_k = \left[\begin{array}{c|ccc} I_k & & & \\ \hline & \frac{1}{x_{k+1}-x_1} & & \\ & & \ddots & \\ & & & \frac{1}{x_n-x_{n-k}} \end{array} \right] \cdot \left[\begin{array}{c|ccc} I_{k-1} & & & \\ \hline & 1 & & \\ & -1 & 1 & \\ & & \ddots & \ddots \\ & & & -1 & 1 \end{array} \right].$$

Moreover, it was proven in [19] (see also [21]) that in many cases the Björck–Pereyra algorithm is guaranteed to provide a very high accuracy.

1.2. Previous work in extensions of this result to polynomial-Vandermonde matrices. The reduction in complexity as well as the favorable numerical properties of the Björck–Pereyra algorithm attracted much attention in the numerical linear algebra literature, and Björck–Pereyra-type algorithms have been derived for several other important classes of *polynomial-Vandermonde* matrices of the form $V_R(x)$ of the form

$$(1.4) \quad V_R(x) = \begin{bmatrix} r_0(x_1) & r_1(x_1) & \cdots & r_{n-1}(x_1) \\ r_0(x_2) & r_1(x_2) & \cdots & r_{n-1}(x_2) \\ \vdots & \vdots & & \vdots \\ r_0(x_n) & r_1(x_n) & \cdots & r_{n-1}(x_n) \end{bmatrix}$$

defined not only by the nodes $\{x_k\}$ but also by the system of polynomials $\{r_k(x)\}$, where the polynomials $R = \{r_k(x)\}$ are assumed to satisfy only the restriction that $\deg r_k(x) = k$. Table 1.1 below lists several classes of polynomials for which the Björck–Pereyra-type algorithms¹ are currently available.

TABLE 1.1
Fast $\mathcal{O}(n^2)$ algorithms for several classes of polynomial-Vandermonde matrices.

Polynomial systems $R = \{r_k(x)\}$	Fast system solver
Monomials	Björck–Pereyra [7]
Chebyshev polynomials	Reichel–Opfer [30]
Real orthogonal (three-term) polynomials	Higham [20]
Szegő polynomials	Bella et al. [3]

In this paper we derive a Björck–Pereyra-type algorithm that applies to the *general polynomial Vandermonde matrices* $V_R(x)$ of the form (1.4). For such a general

¹Along with carrying over the Björck–Pereyra algorithm to various families of polynomials $\{r_k(x)\}$ obeying $\deg r_k(x) = k$, there are also other directions of generalization. For instance, the algorithm had been carried over to the block Vandermonde matrices in [32], to Cauchy matrices in [4], to nonconsecutive powers in [9], and to Bernstein polynomials in [24].

system of polynomials R , however, the matrix $V_R(x)$ is not determined by only $\mathcal{O}(n)$ entries as in the classical case, and so as expected, in this general case the algorithm has the cost of $\mathcal{O}(n^3)$ operations, i.e., it is not fast in this most general case. We show, however, that the derived Björck–Pereyra-type algorithm is fast (again, requiring only $\mathcal{O}(n^2)$ operations, an order of magnitude reduction) when the polynomial-Vandermonde matrices $V_R(x)$ involve a new special class of *quasiseparable* polynomials that generalizes the types of polynomials previously considered.

The key concept in this generalization is the use of *quasiseparable structure* that has received a lot of attention recently in the structured matrices community, mostly for solving various eigenvalue problems. Using quasiseparable structure in the derivation and generalization of Björck–Pereyra-type algorithms is novel.

1.3. Structure of the paper. In the following section, some background material is presented, centering around the use of the confederate matrix corresponding to a system of polynomials. The key concept of the paper, matrices with quasiseparable structure, is also introduced. In section 3, the factorization that is the basis for the new algorithm is presented. In section 4, special cases of the algorithm that have previously been studied are considered as special cases of the new algorithm. In section 5, the class of quasiseparable matrices is defined in depth, and it is shown how properties of this class allow the computational speedup to result in an $\mathcal{O}(n^2)$ overall cost of the new algorithm. A detailed derivation and proof are contained in section 6, some preliminary numerical experiments are presented in section 7, and some conclusions are offered in the final section.

2. Related work and background material.

2.1. Capturing recurrence relations via confederate matrices. To generalize the algorithms in Table 1.1 we will use the concept of a *confederate matrix* introduced in [23]. Let polynomials $R = \{r_0(x), r_1(x), \dots, r_n(x)\}$ be specified by the general recurrence n -term relations²

$$(2.1) \quad r_k(x) = (\alpha_k x - a_{k-1,k}) \cdot r_{k-1}(x) - a_{k-2,k} \cdot r_{k-2}(x) - \dots - a_{0,k} \cdot r_0(x).$$

Define for the polynomial

$$(2.2) \quad \beta(x) = \beta_0 \cdot r_0(x) + \beta_1 \cdot r_1(x) + \dots + \beta_{n-1} \cdot r_{n-1}(x) + r_n(x)$$

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

$$C_R(\beta) = \underbrace{\begin{bmatrix} \frac{a_{01}}{\alpha_1} & \frac{a_{02}}{\alpha_2} & \frac{a_{03}}{\alpha_3} & \dots & \frac{a_{0,k}}{\alpha_k} & \dots & \dots & \frac{a_{0,n}}{\alpha_n} \\ \frac{1}{\alpha_1} & \frac{a_{12}}{\alpha_2} & \frac{a_{13}}{\alpha_3} & \dots & \frac{a_{1,k}}{\alpha_k} & \dots & \dots & \frac{a_{1,n}}{\alpha_n} \\ \frac{1}{\alpha_1} & \frac{1}{\alpha_2} & \frac{a_{23}}{\alpha_3} & \dots & \frac{a_{2,k}}{\alpha_k} & \dots & \dots & \frac{a_{2,n}}{\alpha_n} \\ 0 & \frac{1}{\alpha_2} & \frac{a_{23}}{\alpha_3} & \dots & \frac{a_{k-2,k}}{\alpha_k} & \ddots & \dots & \vdots \\ 0 & 0 & \frac{1}{\alpha_3} & \ddots & \frac{a_{k-1,k}}{\alpha_k} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \frac{a_{k-1,k}}{\alpha_k} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \frac{1}{\alpha_k} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & \dots & \dots & 0 & \frac{1}{\alpha_{n-1}} & \frac{a_{n-1,n}}{\alpha_n} \end{bmatrix}}_{C_R(r_n)}$$

²It is easy to see that any polynomial system $\{r_k(x)\}$ satisfying $\deg r_k(x) = k$ obeys (2.1).

$$(2.3) \quad - \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \vdots \\ \beta_{n-1} \end{bmatrix} \begin{bmatrix} 0 & \cdots & 0 & \frac{1}{\alpha_n} \end{bmatrix}.$$

Notice that the coefficients of the recurrence relations for the k th polynomial $r_k(x)$ from (2.1) are contained in the highlighted k th column of $C_R(r_n)$. We refer to [23] for many useful properties of the confederate matrix and only recall here that

$$r_k(x) = \alpha_0 \cdot \alpha_1 \cdots \alpha_k \cdot \det(xI - [C_R(\beta)]_{k \times k}), \quad \beta(x) = \alpha_0 \cdot \alpha_1 \cdots \alpha_n \cdot \det(xI - C_R(\beta)),$$

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

Next, in Table 2.1 we list confederate matrices for the polynomial systems³ of Table 1.1.

TABLE 2.1
Polynomial systems and corresponding confederate matrices.

Recurrence relations of R	Confederate matrix $C_R(r_n)$
$r_k(x) = x \cdot r_{k-1}(x)$ Monomials	$\begin{bmatrix} 0 & 0 & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix}$ Companion matrix
$r_k(x) = (\alpha_k x - \delta_k) r_{k-1}(x) - \gamma_k r_{k-2}(x)$ Real orthogonal polynomials	$\begin{bmatrix} \frac{\delta_1}{\alpha_1} & \frac{\gamma_2}{\alpha_2} & 0 & \cdots & 0 \\ \frac{1}{\alpha_1} & \frac{\delta_2}{\alpha_2} & \ddots & \ddots & \vdots \\ 0 & \frac{1}{\alpha_2} & \ddots & \frac{\gamma_{n-1}}{\alpha_{n-1}} & 0 \\ \vdots & \ddots & \ddots & \frac{\delta_{n-1}}{\alpha_{n-1}} & \frac{\gamma_n}{\alpha_n} \\ 0 & \cdots & 0 & \frac{1}{\alpha_{n-1}} & \frac{\delta_n}{\alpha_n} \end{bmatrix}$ Tridiagonal matrix
$r_k(x) = \left[\frac{1}{\mu_k} \cdot x + \frac{\rho_k}{\rho_{k-1}} \frac{1}{\mu_k} \right] r_{k-1}(x) - \frac{\rho_k}{\rho_{k-1}} \frac{\mu_{k-1}}{\mu_k} \cdot x \cdot r_{k-2}(x)$ Szegő polynomials	$\begin{bmatrix} -\rho_1 \rho_0^* & \cdots & -\rho_{n-1} \mu_{n-2} \cdots \mu_1 \rho_0^* & -\rho_n \mu_{n-1} \cdots \mu_1 \rho_0^* \\ \mu_1 & \ddots & -\rho_{n-1} \mu_{n-2} \cdots \mu_2 \rho_1^* & -\rho_n \mu_{n-1} \cdots \mu_2 \rho_1^* \\ 0 & \ddots & \vdots & \vdots \\ \vdots & & & \\ 0 & \cdots & -\rho_{n-1} \rho_{n-2}^* & -\rho_n \mu_{n-1} \rho_{n-2}^* \\ & & \mu_{n-1} & -\rho_n \rho_{n-1}^* \end{bmatrix}$ Unitary Hessenberg matrix

It turns out that tridiagonal and unitary Hessenberg matrices of Table 2.1 are special cases of the more general class of matrices defined next.

³For the monomials and for the real orthogonal polynomials, the structure of the confederate matrices can be immediately deduced from their recurrence relations. For Szegő polynomials it is also well known, see; e.g., [26] and the references therein.

2.2. Hessenberg quasiseparable matrices and polynomials. Recall that a matrix $A = [a_{ij}]$ is called *upper Hessenberg* if all entries below the first subdiagonal are zeros; that is, $a_{ij} = 0$ if $i > j + 1$, and is furthermore *strongly upper Hessenberg* provided none of the subdiagonal elements are zeros, so $a_{i+1,i} \neq 0$ for $i = 1, \dots, n-1$.

DEFINITION 2.1 (Quasiseparable matrices and polynomials).

- A matrix A is called (H, m) -quasiseparable if **(i)** it is strongly upper Hessenberg, and **(ii)** $\max(\text{rank } A_{12}) = m$ where the maximum is taken over all symmetric partitions of the form

$$(2.4) \quad A = \left[\begin{array}{c|c} * & A_{12} \\ * & * \end{array} \right];$$

for instance, the low-rank blocks of a 5×5 (H, m) -quasiseparable matrix would be those shaded below:

$$\begin{array}{c} \left[\begin{array}{c|ccccc} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{array} \right] \quad \left[\begin{array}{cc|ccc} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{array} \right] \\ \\ \left[\begin{array}{ccc|cc} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{array} \right] \quad \left[\begin{array}{cccc|c} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{array} \right]. \end{array}$$

- Let $A = [a_{ij}]$ be an (H, m) -quasiseparable matrix. Then the system of polynomials $\{r_k(x)\}$ related to A via

$$r_k(x) = \alpha_1 \cdots \alpha_k \det(xI - A)_{(k \times k)} \quad (\text{where } \alpha_i = 1/a_{i+1,i})$$

is called a system of (H, m) -quasiseparable polynomials. That is, using the terminology of the previous section, (H, m) -quasiseparable polynomials are those polynomials with an (H, m) -quasiseparable confederate matrix.

Next, brief verifications are presented that the class of $(H, 1)$ -quasiseparable polynomials is wide enough to include monomials and real orthogonal and Szegő polynomials (i.e., all polynomials of Tables 1.1 and 2.1) as special cases. This can be seen by inspecting, for each confederate matrix, its typical submatrix A_{12} from the partition described in (2.4).

2.3. Special cases of (H, m) -quasiseparable polynomials. In this section, several examples of (H, m) -quasiseparable polynomials are given, demonstrating that while covering old cases, the class of (H, m) -quasiseparable polynomials also includes new cases not considered by any previous work.

2.3.1. Real orthogonal polynomials. To demonstrate this fact, we must show that the confederate matrix to which real orthogonal polynomials are related is $(H, 1)$ -quasiseparable. From Table 2.1, we have that real orthogonal polynomials have a tridiagonal confederate matrix, denoting this tridiagonal matrix A , then the submatrix A_{12} has the form $(\delta_k/\alpha_k)e_m e_1^T$, which can easily be observed to have rank one.

2.3.2. Szegő polynomials. Similar to the above section, from Table 2.1 we have that Szegő polynomials are related to unitary Hessenberg confederate matrices. Denoting such a matrix by A , then the corresponding submatrix A_{12} has the form

$$A_{12} = \begin{bmatrix} -\rho_k \mu_{k-1} \cdots \mu_3 \mu_2 \mu_1 \rho_0^* & -\rho_{k-1} \mu_{k-2} \cdots \mu_3 \mu_2 \mu_1 \rho_0^* & \cdots & -\rho_n \mu_{n-1} \cdots \mu_3 \mu_2 \mu_1 \rho_0^* \\ -\rho_k \mu_{k-1} \cdots \mu_3 \mu_2 \rho_1^* & -\rho_{k-1} \mu_{k-2} \cdots \mu_3 \mu_2 \rho_1^* & \cdots & -\rho_n \mu_{n-1} \cdots \mu_3 \mu_2 \rho_1^* \\ -\rho_k \mu_{k-1} \cdots \mu_3 \rho_2^* & -\rho_{k-1} \mu_{k-2} \cdots \mu_3 \rho_2^* & \cdots & -\rho_n \mu_{n-1} \cdots \mu_3 \rho_2^* \end{bmatrix},$$

which is also rank 1 since the rows are scalar multiples of each other.

2.3.3. m -recurrent polynomials. It is easy to see that if polynomials satisfy m -term recurrence relations

$$(2.5) \quad r_k(x) = (\alpha_k x - a_{k-1,k}) \cdot r_{k-1}(x) - a_{k-2,k} \cdot r_{k-2}(x) - \cdots - a_{k-(m-1),k} \cdot r_{k-(m-1)}(x),$$

then their confederate matrices

$$(2.6) \quad A = \begin{bmatrix} \frac{a_{0,1}}{\alpha_1} & \cdots & \frac{a_{0,m-1}}{\alpha_{m-1}} & 0 & \cdots & 0 \\ \frac{1}{\alpha_1} & \frac{a_{1,2}}{\alpha_2} & \cdots & \frac{a_{1,m}}{\alpha_m} & \ddots & \vdots \\ 0 & \frac{1}{\alpha_2} & & & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & & \frac{a_{n-(m-1),n}}{\alpha_n} \\ \vdots & & \ddots & \frac{1}{\alpha_{n-2}} & & \vdots \\ 0 & \cdots & \cdots & 0 & \frac{1}{\alpha_{n-1}} & \frac{a_{n-1,n}}{\alpha_n} \end{bmatrix}$$

are $(1, m-2)$ -banded, i.e., they have only one nonzero subdiagonal and $m-2$ nonzero superdiagonals. Clearly, any A_{12} in (2.4) has rank at most $(m-2)$, implying that A is a $(H, m-2)$ -quasiseparable matrix by definition. Hence m -recurrent polynomials are $(H, m-2)$ -quasiseparable.

In particular, when $m=3$, the resulting system of polynomials are real orthogonal polynomials (as they are 3-recurrent, or satisfy three-term recurrence relations), and these results imply that the related confederate matrix, tridiagonal, is $(H, 3-2)$ - or $(H, 1)$ -quasiseparable, as described in section 2.3.1.

2.3.4. Polynomials satisfying more general three-term recurrence relations. Consider polynomials satisfying fairly general⁴ three-term recurrence relations

$$(2.7) \quad r_k(x) = (\alpha_k x - \delta_k) \cdot r_{k-1}(x) - (\beta_k x + \gamma_k) \cdot r_{k-2}(x).$$

It was observed in [2] that the confederate matrix $C_R(r_n)$ of such $\{r_k(x)\}$ has the

⁴That is more general than three-term recurrence relations of Table 2.1.

form

$$(2.8) \quad \begin{bmatrix} \frac{\delta_1}{\alpha_1} & \frac{\delta_1 \beta_2 + \gamma_2}{\alpha_2} & \frac{\delta_1 \beta_2 + \gamma_2}{\alpha_2} \left(\frac{\beta_3}{\alpha_3} \right) & \frac{\delta_1 \beta_2 + \gamma_2}{\alpha_2} \left(\frac{\beta_3}{\alpha_3} \right) \left(\frac{\beta_4}{\alpha_4} \right) & \cdots & \frac{\delta_1 \beta_2 + \gamma_2}{\alpha_2} \left(\frac{\beta_3}{\alpha_3} \right) \left(\frac{\beta_4}{\alpha_4} \right) \cdots \left(\frac{\beta_n}{\alpha_n} \right) \\ \frac{1}{\alpha_1} & \frac{\delta_2}{\alpha_2} + \frac{\beta_2}{\alpha_1 \alpha_2} & \frac{\left(\frac{\delta_2}{\alpha_2} + \frac{\beta_2}{\alpha_1 \alpha_2} \right) \beta_3 + \gamma_3}{\alpha_3} & \frac{\left(\frac{\delta_2}{\alpha_2} + \frac{\beta_2}{\alpha_1 \alpha_2} \right) \beta_3 + \gamma_3}{\alpha_3} \left(\frac{\beta_4}{\alpha_4} \right) & \cdots & \frac{\left(\frac{\delta_2}{\alpha_2} + \frac{\beta_2}{\alpha_1 \alpha_2} \right) \beta_3 + \gamma_3}{\alpha_3} \left(\frac{\beta_4}{\alpha_4} \right) \cdots \left(\frac{\beta_n}{\alpha_n} \right) \\ & \frac{1}{\alpha_2} & \frac{\delta_3}{\alpha_3} + \frac{\beta_3}{\alpha_2 \alpha_3} & \frac{\left(\frac{\delta_3}{\alpha_3} + \frac{\beta_3}{\alpha_2 \alpha_3} \right) \beta_4 + \gamma_4}{\alpha_4} & & \\ & & \frac{1}{\alpha_3} & \frac{\delta_4}{\alpha_4} + \frac{\beta_4}{\alpha_3 \alpha_4} & & \vdots \\ & & & \frac{1}{\alpha_4} & & \vdots \\ & & & & \ddots & \\ & & & & & \frac{\left(\frac{\delta_{n-1}}{\alpha_{n-1}} + \frac{\beta_{n-1}}{\alpha_{n-2} \alpha_{n-1}} \right) \beta_n + \gamma_n}{\alpha_n} \\ & & & & & \frac{1}{\alpha_{n-1}} & \frac{\delta_n}{\alpha_n} + \frac{\beta_n}{\alpha_{n-1} \alpha_n} \end{bmatrix}.$$

Observe that by taking $\beta_k = 0$ for each k , the matrix A of (2.8) reduces to the tridiagonal matrix displayed in Table 2.1. Second, inserting the relations

$$\alpha_k = \frac{1}{\mu_k}, \quad \delta_k = -\frac{1}{\mu_k} \frac{\rho_k}{\rho_{k-1}}, \quad \beta_k = \frac{\mu_{k-1}}{\mu_k} \frac{\rho_k}{\rho_{k-1}}, \quad \gamma_k = 0$$

into the matrix A of (2.8) results in the unitary Hessenberg matrix displayed in Table 2.1.

It is easy to verify that the matrix A of (2.8) is irreducible $(H, 1)$ -quasiseparable. Indeed, one can see that in any partition (2.4) the $(k-1)$ st column⁵ $A_{12}(:, k-1)$ and the k th column $A_{12}(:, k)$ of the matrix A_{12} are scalar multiples of each other:

$$A_{12}(:, k) = \frac{\beta_{k+2}}{\alpha_{k+2}} A_{12}(:, k-1).$$

For example, inspect the $2 \times (n-2)$ matrix

$$A_{12} = \begin{bmatrix} \frac{\delta_1 \beta_2 + \gamma_2}{\alpha_2} \left(\frac{\beta_3}{\alpha_3} \right) & \frac{\delta_1 \beta_2 + \gamma_2}{\alpha_2} \left(\frac{\beta_3}{\alpha_3} \right) \left(\frac{\beta_4}{\alpha_4} \right) & \frac{\delta_1 \beta_2 + \gamma_2}{\alpha_2} \left(\frac{\beta_3}{\alpha_3} \right) \left(\frac{\beta_4}{\alpha_4} \right) \left(\frac{\beta_5}{\alpha_5} \right) \cdots \\ \frac{\left(\frac{\delta_2}{\alpha_2} + \frac{\beta_2}{\alpha_1 \alpha_2} \right) \beta_3 + \gamma_3}{\alpha_3} & \frac{\left(\frac{\delta_2}{\alpha_2} + \frac{\beta_2}{\alpha_1 \alpha_2} \right) \beta_3 + \gamma_3}{\alpha_3} \left(\frac{\beta_4}{\alpha_4} \right) & \frac{\left(\frac{\delta_2}{\alpha_2} + \frac{\beta_2}{\alpha_1 \alpha_2} \right) \beta_3 + \gamma_3}{\alpha_3} \left(\frac{\beta_4}{\alpha_4} \right) \left(\frac{\beta_5}{\alpha_5} \right) \cdots \end{bmatrix}.$$

Hence, polynomials (2.7) are $(H, 1)$ -quasiseparable.

2.3.5. Polynomials satisfying Szegő-type two-term recurrence relations.

Consider polynomials $\{r_k(x)\}$ satisfying general two-term recurrence relations of the Szegő type,

$$(2.9) \quad \begin{bmatrix} G_k(x) \\ r_k(x) \end{bmatrix} = \begin{bmatrix} \alpha_k & \beta_k \\ \gamma_k & 1 \end{bmatrix} \begin{bmatrix} G_{k-1}(x) \\ (\delta_k x + \theta_k) r_{k-1}(x) \end{bmatrix}.$$

Here $\{G_k(x)\}$ are some auxiliary polynomials. The class of polynomials (2.9) includes the classical Szegő polynomials $\{r_k(x)\}$ satisfying⁶

$$\begin{bmatrix} G_k(x) \\ r_k(x) \end{bmatrix} = \frac{1}{\mu_k} \begin{bmatrix} 1 & -\rho_k \\ -\rho_k^* & 1 \end{bmatrix} \begin{bmatrix} G_{k-1}(x) \\ x r_{k-1}(x) \end{bmatrix}.$$

⁵The MATLAB notation $A(i : j, k : l)$ denotes the submatrix obtained from rows $i, i+1, \dots, j$ and columns $k, k+1, \dots, l$, and a lone colon denotes the entire row or column.

⁶Here the complex numbers $|\rho_k| \leq 1$ are referred to as *reflection coefficients*, and $\mu_k := \sqrt{1 - |\rho_k|^2}$ if $|\rho_k| < 1$ and $\mu_k := 1$ if $|\rho_k| = 1$ are called *complementary parameters*.

It was shown in [2] that the confederate matrix $C_R(r_n)$ of $\{r_k(x)\}$ satisfying (2.9) has the form

$$(2.10) \quad \begin{bmatrix} -\frac{\theta_1+\gamma_1}{\delta_1} & -(\alpha_1-\beta_1\gamma_1)\frac{\gamma_2}{\delta_2} & -(\alpha_1-\beta_1\gamma_1)(\alpha_2-\beta_2\gamma_2)\frac{\gamma_3}{\delta_3} & \cdots & -(\alpha_1-\beta_1\gamma_1)\cdots(\alpha_{n-1}-\beta_{n-1}\gamma_{n-1})\frac{\gamma_n}{\delta_n} \\ \frac{1}{\delta_1} & -\frac{\theta_2+\gamma_2\beta_1}{\delta_2} & -\beta_1(\alpha_2-\beta_2\gamma_2)\frac{\gamma_3}{\delta_3} & \cdots & -\beta_1(\alpha_2-\beta_2\gamma_2)\cdots(\alpha_{n-1}-\beta_{n-1}\gamma_{n-1})\frac{\gamma_n}{\delta_n} \\ 0 & \frac{1}{\delta_2} & -\frac{\theta_3+\gamma_3\beta_2}{\delta_3} & \ddots & -\beta_2(\alpha_3-\beta_3\gamma_3)\cdots(\alpha_{n-1}-\beta_{n-1}\gamma_{n-1})\frac{\gamma_n}{\delta_n} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & -\beta_{n-1}(\alpha_{n-1}-\beta_{n-1}\gamma_{n-1})\frac{\gamma_n}{\delta_n} \\ 0 & \cdots & 0 & \frac{1}{\delta_{n-1}} & -\frac{\theta_n+\gamma_n\beta_{n-1}}{\delta_n} \end{bmatrix}.$$

As in the unitary Hessenberg case, it is easy to see that the rows of A_{12} are scalar multiples of each other and hence A is $(H, 1)$ -quasiseparable, and the polynomials $\{r_k(x)\}$ in (2.9) are $(H, 1)$ -quasiseparable.

2.3.6. Polynomials satisfying EGO-type two-term recurrence relations.

Finally, suppose the polynomials $\{r_k(x)\}$ satisfy the recurrence relations

$$(2.11) \quad \begin{bmatrix} G_k(x) \\ r_k(x) \end{bmatrix} = \begin{bmatrix} \alpha_k & \beta_k \\ \gamma_k & \delta_k x + \theta_k \end{bmatrix} \begin{bmatrix} G_{k-1}(x) \\ r_{k-1}(x) \end{bmatrix},$$

similar to those derived in [13]. Again, $\{G_k(x)\}$ are some auxiliary polynomials. It was shown in [2] that the confederate matrix $C_R(r_n)$ of such $\{r_k(x)\}$ has the form

$$(2.12) \quad \begin{bmatrix} -\frac{\theta_1}{\delta_1} & -\beta_1(\frac{\gamma_2}{\delta_2}) & -\beta_1\alpha_2(\frac{\gamma_3}{\delta_3}) & -\beta_1\alpha_2\alpha_3(\frac{\gamma_4}{\delta_4}) & \cdots & -\beta_1\alpha_2\alpha_3\alpha_4\cdots\alpha_{n-1}(\frac{\gamma_n}{\delta_n}) \\ \frac{1}{\delta_1} & -\frac{\theta_2}{\delta_2} & -\beta_2(\frac{\gamma_3}{\delta_3}) & -\beta_2\alpha_3(\frac{\gamma_4}{\delta_4}) & \cdots & -\beta_2\alpha_3\alpha_4\cdots\alpha_{n-1}(\frac{\gamma_n}{\delta_n}) \\ 0 & \frac{1}{\delta_2} & -\frac{\theta_3}{\delta_3} & -\beta_3(\frac{\gamma_4}{\delta_4}) & \ddots & -\beta_3\alpha_4\cdots\alpha_{n-1}(\frac{\gamma_n}{\delta_n}) \\ 0 & 0 & \frac{1}{\delta_3} & -\frac{\theta_4}{\delta_4} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & -\beta_{n-1}(\frac{\gamma_n}{\delta_n}) \\ 0 & \cdots & 0 & 0 & \frac{1}{\delta_{n-1}} & -\frac{\theta_n}{\delta_n} \end{bmatrix}.$$

Again, a straightforward inspection of the corresponding A_{12} indicates that polynomials $\{r_k(x)\}$ of (2.11) form a $(H, 1)$ -quasiseparable system.

To sum up, the class of (H, m) -quasiseparable polynomials includes not only the well-studied classes of real orthogonal polynomials and the Szegő polynomials, but also several other interesting polynomial classes described in sections 2.3.3–2.3.6. Hence, it is of interest to generalize the Björck–Pereyra algorithm to a polynomial-Vandermonde matrix $V_R(x)$ corresponding to a system of (H, m) -quasiseparable polynomials R . This is exactly what is done in the rest of the paper.

3. New Björck–Pereyra-type algorithm. General Hessenberg case.

In this section we consider the linear system $V_R(x)a = f$, where $V_R(x)$ is the polynomial-Vandermonde matrix corresponding to the polynomial system R and the n distinct nodes x . No restrictions are placed on the polynomial system R at this point other than $\deg(r_k) = k$. The new algorithm is based on a decomposition of the inverse $V_R(x)^{-1}$ enabled by the following lemma. Herein we use the MATLAB convention $x_{i:j} = [x_i \ x_{i+1} \ \cdots \ x_{j-1} \ x_j]^T$ to denote a portion of the larger vector $x = [x_1 \ x_2 \ \cdots \ x_{n-1} \ x_n]^T$.

LEMMA 3.1. *Let $R = \{r_0(x), \dots, r_n(x)\}$ be an arbitrary system of polynomials as in (2.1), and denote $R_1 = \{r_0(x), \dots, r_{n-1}(x)\}$. Further let $x_{1:n} = (x_1, \dots, x_n)$ be n distinct points. Then the inverse of $V_R(x_{1:n})$ admits a decomposition*

$$(3.1) \quad V_R(x_{1:n})^{-1} = U_1 \cdot \begin{bmatrix} 1 & 0 \\ 0 & V_R(x_{2:n})^{-1} \end{bmatrix} L_1,$$

with

$$(3.2) \quad U_1 = \begin{bmatrix} \frac{1}{\alpha_0} \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{bmatrix} C_{R_1}(r_{n-1}) - x_1 I \\ \dots & 0 & \frac{1}{\alpha_{n-1}} \end{bmatrix},$$

$$(3.3) \quad L_1 = \begin{bmatrix} 1 & & & \\ & \frac{1}{x_2 - x_1} & & \\ & & \ddots & \\ & & & \frac{1}{x_n - x_1} \end{bmatrix} \begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ \vdots & & \ddots & \\ -1 & & & 1 \end{bmatrix}.$$

The proof of Lemma 3.1 is given in section 6, but first we present its use in solving the linear system $V_R(x)a = f$.

3.1. Solving polynomial Vandermonde systems. Like the classical Björck–Pereyra algorithm, the recursive nature of (3.1) allows a decomposition of $V_R(x)^{-1}$ into $2n - 2$ factors,

$$(3.4) \quad V_R(x)^{-1} = U_1 \cdot \left[\begin{array}{c|c} I_1 & \\ \hline & U_2 \end{array} \right] \cdots \left[\begin{array}{c|c} I_{n-2} & \\ \hline & U_{n-1} \end{array} \right] \cdot \left[\begin{array}{c|c} I_{n-2} & \\ \hline & L_{n-1} \end{array} \right] \cdots \left[\begin{array}{c|c} I_1 & \\ \hline & L_2 \end{array} \right] \cdot L_1,$$

with the lower and upper triangular factors given in (3.2), (3.3). The associated linear system can be solved by multiplying (3.4) by the right-hand side vector f .

It is emphasized that this decomposition is valid for any polynomial system R ; however, no computational savings are guaranteed. In order to have the desired computational savings, each multiplication of a matrix from (3.4) by a vector must be performed quickly.

The factors L_k are sparse as in the classical Björck–Pereyra algorithm, and thus multiplication by them is fast. However, unlike the classical Björck–Pereyra algorithm, the factors U_k are not sparse in general. In order to have a fast $\mathcal{O}(n^2)$ algorithm for solving the system $V_R(x)a = f$, it is necessary to be able to multiply each matrix in (3.4) by f in $\mathcal{O}(n)$ operations.

3.2. Differences between L_k and \tilde{L}_k . It is easy to see that the classical Vandermonde matrix V involving monomials and the polynomial Vandermonde matrix V_R are related via

$$V_R = V \cdot U,$$

where U is a change of basis matrix:

$$\begin{bmatrix} 1 & x & \cdots & x^{n-1} \end{bmatrix} \cdot U = \begin{bmatrix} r_0(x) & r_1(x) & \cdots & r_{n-1}(x) \end{bmatrix}.$$

Since U is upper triangular and since the LU factorization is unique, it follows that both matrices V and V_R share the same L factor that can be equivalently factored in different ways, e.g., using elementary factors (1.3) as in (1.1) or using elementary factors (3.3) as in (3.4).

Our choice of (3.3) is motivated by the heuristics that the latter uses the numbers $\{(x_2 - x_1)^{-1}, \dots, (x_n - x_1)^{-1}\}$. Since the same x_1 is used in each factor, (3.3) can potentially provide better accuracy if the so-called Leja ordering (that maximizes the separation between x_1 to the other nodes) is used [20, 27, 30]. See section 7 for more details.

4. Known special cases where the Björck–Pereyra-type algorithm is fast. We next present a detailed reduction of the algorithm presented in the previous section in several important special cases that were studied earlier by different authors.

4.1. Monomials. The classical Björck–Pereyra algorithm. Suppose the system of polynomials in the polynomial-Vandermonde matrix is simply a set of monomials; that is, $R = \{1, x, \dots, x^{n-1}, x^n\}$. Then (2.1) becomes simply

$$(4.1) \quad r_0(x) = 1, \quad r_k(x) = x r_{k-1}(x), \quad k = 1, \dots, n$$

and the corresponding confederate matrix is

$$(4.2) \quad C_R(r_n) = \begin{bmatrix} 0 & 0 & \cdots & \cdots & 0 \\ 1 & \ddots & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix};$$

that is, $\alpha_k = 1$ for $k = 0, \dots, n-1$. Inserting this and (4.2) into (3.2) yields (1.2), implying the factors U_k reduce to those of the classical Björck–Pereyra algorithm in this case. That is, the Vandermonde linear system $V_R(x)a = f$ can be solved via the factorization of (3.4):

$$(4.3) \quad a = V_R(x)^{-1}f = U_1 \cdot \left[\begin{array}{c|c} I_1 & \\ \hline & U_2 \end{array} \right] \cdots \left[\begin{array}{c|c} I_{n-2} & \\ \hline & U_{n-1} \end{array} \right] \\ \cdot \left[\begin{array}{c|c} I_{n-2} & \\ \hline & L_{n-1} \end{array} \right] \cdots \left[\begin{array}{c|c} I_1 & \\ \hline & L_2 \end{array} \right] \cdot L_1 \cdot f.$$

Thus when the quasiseparable polynomials in question reduce to the monomials, the algorithm described reduces to the classical Björck–Pereyra algorithm. Due to the sparseness of the matrices involved, the overall cost of the algorithm is only $\frac{5}{2}n^2$.

4.2. Real orthogonal polynomials. The Higham algorithm. If the polynomial system under consideration satisfies the three-term recurrence relations

$$(4.4) \quad r_k(x) = (\alpha_k x - \delta_k)r_{k-1}(x) - \gamma_k r_{k-2}(x),$$

then the resulting confederate matrix is tridiagonal

$$(4.5) \quad C_R(r_n) = \begin{bmatrix} \frac{\delta_1}{\alpha_1} & \frac{\gamma_2}{\alpha_2} & 0 & \cdots & 0 \\ \frac{1}{\alpha_1} & \frac{\delta_2}{\alpha_2} & \ddots & \ddots & \vdots \\ 0 & \frac{1}{\alpha_2} & \ddots & \frac{\gamma_{n-1}}{\alpha_{n-1}} & 0 \\ \vdots & & \ddots & \frac{\delta_{n-1}}{\alpha_{n-1}} & \frac{\gamma_n}{\alpha_n} \\ 0 & \cdots & 0 & \frac{1}{\alpha_{n-1}} & \frac{\delta_n}{\alpha_n} \end{bmatrix},$$

which leads to the matrices U_k of the form

$$U_k = \begin{bmatrix} \frac{1}{\alpha_0} & \frac{\delta_1}{\alpha_1} - x_k & \frac{\gamma_2}{\alpha_2} & 0 & \cdots & 0 \\ 0 & \frac{1}{\alpha_1} & \frac{\delta_2}{\alpha_2} - x_k & \ddots & \ddots & \vdots \\ 0 & \frac{1}{\alpha_2} & \ddots & \frac{\gamma_{k-1}}{\alpha_{k-1}} & & 0 \\ \vdots & \vdots & & \ddots & \frac{\delta_{k-1}}{\alpha_{k-1}} - x_k & \frac{\gamma_k}{\alpha_k} \\ 0 & \cdots & 0 & \frac{1}{\alpha_{k-1}} & \frac{\delta_k}{\alpha_k} - x_k & \\ 0 & \cdots & & 0 & \frac{1}{\alpha_{n-j}} & \end{bmatrix}.$$

Again, the factorization (4.3) uses the above matrix to solve the linear system. The sparseness of these matrices allows computational savings, and the overall cost of the algorithm is again $\mathcal{O}(n^2)$.

In this case, the entire algorithm presented reduces to Algorithm 2.1 in [20]. In particular, the multiplication of a vector by the matrices specified in (4.3) involving L_k and U_k can be seen as stage I and stage II in that algorithm, respectively.

4.3. Szegő polynomials. The [3] algorithm. If the system of quasiseparable polynomials are the Szegő polynomials $\Phi^\# = \{\phi_0^\#(x), \dots, \phi_n^\#(x)\}$ represented by the reflection coefficients ρ_k and complementary parameters μ_k (see [1]), then they satisfy the recurrence relations

$$\phi_0^\#(x) = 1, \quad \phi_1^\#(x) = \frac{1}{\mu_1} \cdot x \phi_0^\#(x) - \frac{\rho_1}{\mu_1} \phi_0^\#(x)$$

$$(4.6) \quad \phi_k^\#(x) = \left[\frac{1}{\mu_k} \cdot x + \frac{\rho_k}{\rho_{k-1}} \frac{1}{\mu_k} \right] \phi_{k-1}^\#(x) - \frac{\rho_k}{\rho_{k-1}} \frac{\mu_{k-1}}{\mu_k} \cdot x \cdot \phi_{k-2}^\#(x)$$

which are known to be associated to the almost unitary Hessenberg matrix

$$(4.7) \quad C_{\Phi^\#}(\phi_n^\#) = \begin{bmatrix} -\rho_1 \rho_0^* & \cdots & -\rho_{n-1} \mu_{n-2} \cdots \mu_1 \rho_0^* & -\rho_n \mu_{n-1} \cdots \mu_1 \rho_0^* \\ \mu_1 & \ddots & -\rho_{n-1} \mu_{n-2} \cdots \mu_2 \rho_1^* & -\rho_n \mu_{n-1} \cdots \mu_2 \rho_1^* \\ 0 & \ddots & \vdots & \vdots \\ \vdots & & -\rho_{n-1} \rho_{n-2}^* & -\rho_n \mu_{n-1} \rho_{n-2}^* \\ 0 & \cdots & \mu_{n-1} & -\rho_n \rho_{n-1}^* \end{bmatrix}.$$

In particular, if (4.7) is inserted into the factors (3.2) in (4.3), then the result is exactly that derived in [3, equations (3.10) and (3.15)], where the nice properties of

the matrix $C_{\Phi\#}(\phi_n^\#)$ were used to provide a computational speedup. Specifically, the algorithm is made fast by the factorization

$$(4.8) \quad C_{\Phi\#}(\phi_n^\#) = G(\rho_1) \cdot G(\rho_2) \cdot \cdots \cdot G(\rho_{n-1}) \cdot \tilde{G}(\rho_n),$$

where

$$G(\rho_j) = \text{diag} \left\{ I_{j-1}, \begin{bmatrix} \rho_j & \mu_j \\ \mu_j & -\rho_j^* \end{bmatrix}, I_{n-j-1} \right\}, \quad j = 1, 2, \dots, n-1$$

and

$$\tilde{G}(\rho_n) = \text{diag}\{I_{n-1}, \rho_n\};$$

see, for instance, [1, 14, 29]. This gives an overall computational cost of $\mathcal{O}(n^2)$.

In the next section, we present a new special case which contains all previous special cases.

5. A new special case. (H, m) -quasiseparable polynomials.

5.1. Rank definition. As stated in the introduction, a matrix A is called upper quasiseparable of order m if $\max(\text{rank}A_{12}) = m$, where the maximum is taken over all symmetric partitions of the form

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

Also, a matrix $A = [a_{ij}]$ is called *upper Hessenberg* if all entries below the first subdiagonal are zeros; that is, $a_{ij} = 0$ if $i > j + 1$. For brevity, we shall refer to order m upper quasiseparable Hessenberg matrices as (H, m) -quasiseparable matrices. Polynomials corresponding to (H, m) -quasiseparable matrices are called (H, m) -quasiseparable polynomials.

5.2. Generator definition. We next present an equivalent definition of an (H, m) -quasiseparable matrix in terms of its *generators*. The equivalence of these two definitions is well known, see, e.g., in [10]. An $n \times n$ matrix $C_R(r_n)$ is called (H, m) -quasiseparable if it is of the form

$$C_R(r_n) = \begin{array}{|c|} \hline \begin{array}{c} d_1 \\ p_2 q_1 \\ \vdots \\ \vdots \\ 0 \end{array} \\ \hline \end{array} \begin{array}{c} g_i b_{ij}^\times h_j \\ \vdots \\ \vdots \\ p_n q_{n-1} \\ d_n \end{array}$$

with

$$(5.1) \quad b_{ij}^\times = (b_{i+1}) \cdots (b_{j-1}), \quad b_{i,i+1} = I.$$

Here p_k, q_k, d_k are scalars, the elements g_k are row vectors of maximal size m , h_k are column vectors of maximal size m , and b_k are matrices of maximal size $m \times m$ such that all products make sense. The elements $\{p_k, q_k, d_k, g_k, b_k, h_k\}$ are called the *generators* of the matrix $C_R(r_n)$.

The elements in the upper part of the matrix $g_i b_{ij}^\times h_j$ are products of a row vector, a (possibly empty) sequence of matrices possibly of different sizes, and finally a column vector, as depicted here:

$$(5.2) \quad g_i b_{ij}^\times h_j = \overbrace{g_i}^{1 \times u_i} \overbrace{b_{i+1}}^{u_i \times u_{i+1}} \overbrace{b_{i+2}}^{u_{i+1} \times u_{i+2}} \cdots \overbrace{b_{j-1}}^{u_{j-2} \times u_{j-1}} \overbrace{h_j}^{u_{j-1} \times 1}$$

with $u_k \leq m$ for each $k = 1, \dots, n - 1$.

5.3. Fast multiplication using the quasiseparable structure. Here we show that in the special case of (H, m) -quasiseparable polynomials our Björck–Pereyra-type algorithms is fast, requiring $\mathcal{O}(n^2)$ operations. In view of (4.3) it suffices to have an algorithm for $\mathcal{O}(n)$ multiplication of a quasiseparable matrix by a vector since each matrix U_k contains a quasiseparable matrix as in (3.2). With such an algorithm, each multiplication in (3.4) could be implemented in $\mathcal{O}(n)$ operations, hence the total cost of computing the solution a would be $\mathcal{O}(n^2)$. The fast multiplication algorithm presented next is valid for a slightly more general class of quasiseparable matrices; specifically, the Hessenberg structure emphasized above is not essential here. To describe a more general result, we first give a slightly more general definition.

A matrix A is called (n_L, n_U) -quasiseparable if, for integers n_L and n_U , $\max(\text{rank} A_{12}) = n_U$ and $\max(\text{rank} A_{21}) = n_L$, where the maximum is taken over all symmetric partitions of the form

$$(5.3) \quad A = \left[\begin{array}{c|c} * & A_{12} \\ \hline A_{21} & * \end{array} \right].$$

Similar to the generator representation for upper quasiseparable matrices given above, arbitrary order quasiseparable matrices can be expressed in terms of generators as

$$C_R(r_n) = \begin{array}{|c|} \hline \begin{array}{c} d_1 \\ \vdots \\ \vdots \\ \vdots \\ d_n \end{array} \\ \hline \end{array} \begin{array}{c} g_i b_{ij}^\times h_j \\ \vdots \\ \vdots \\ \vdots \\ p_i a_{ij}^\times q_j \end{array}$$

with $a_{ij}^\times = (a_{i-1}) \cdots (a_{j+1})$, $a_{i+1,i} = I$, b_{ij} is as defined in (5.1). Now d_k are scalars, the elements p_k, g_k are row vectors of maximal size n_L and n_U , respectively, q_k, h_k are column vectors of maximal size n_L and n_U , respectively, and a_k, b_k are matrices of maximal size $n_L \times n_L$ and $n_U \times n_U$, respectively, such that all products make sense. The entries in the lower triangular part are defined by the product $p_i a_{ij}^\times q_j$, which has a similar form to that shown in (5.2).

Such a fast algorithm for multiplying a quasiseparable matrix by a vector is suggested by the following decomposition, valid for any (n_L, n_U) -quasiseparable matrix.

PROPOSITION 5.1. *Let $C_R(r_n)$ be an $n \times n$ (n_L, n_U) -quasiseparable matrix specified by its generators as in section 5.2. Then $C_R(r_n)$ admits the decomposition*

$$C_R(r_n) = L + D + U,$$

where

$$L = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & p_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & p_n \end{bmatrix} \begin{bmatrix} 0 & \cdots & 0 & 0 \\ \hline \tilde{A}^{-1} & \vdots & \vdots & 0 \\ 0 & \vdots & \vdots & 0 \end{bmatrix} \begin{bmatrix} q_1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & q_{n-1} & 0 \\ 0 & \cdots & 0 & 1 \end{bmatrix},$$

$$D = \begin{bmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & d_n \end{bmatrix},$$

$$(5.4) \quad U = \begin{bmatrix} g_1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & g_{n-1} & 0 \\ 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & \vdots & \vdots & 0 \\ \hline \tilde{B}^{-1} & \vdots & \vdots & 0 \\ 0 & \cdots & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & h_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & h_n \end{bmatrix},$$

$$= \begin{bmatrix} I_{n_1} & b_1 & b_1 b_2 & b_1 b_2 b_3 & \cdots & b_1 b_2 b_3 \cdots b_{m-1} \\ & I_{n_2} & b_2 & b_2 b_3 & \cdots & b_2 b_3 \cdots b_{m-1} \\ & & \ddots & \ddots & & \vdots \\ & & & I_{n_{m-2}} & b_{m-2} & b_{m-2} b_{m-1} \\ & & & & I_{n_{m-1}} & b_{m-1} \\ & & & & & I_{n_m} \end{bmatrix}.$$

We next consider the computational cost of the multiplication algorithm suggested by this proposition. Let $C_R(r_n)$ be a quasiseparable matrix of order (n_L, m) whose generators are of the following sizes:

Generator	p_k	a_k	q_k	d_k	g_k	b_k	h_k
Size	$1 \times l_{k-1}$	$l_k \times l_{k-1}$	$l_k \times 1$	1×1	$1 \times u_k$	$u_{k-1} \times u_k$	$u_{k-1} \times 1$

for numbers $l_k, u_k, k = 1, \dots, n$. Define also $l_0 = u_0 = 0$. Then it can be seen that the computational cost is

$$c = 3n + 2 \sum_{k=1}^{n-1} (u_k + l_k + u_k u_{k-1} + l_k l_{k-1})$$

flops (additions plus multiplications). Since $l_k \leq n_L$ and $u_k \leq n_U$ for $k = 1, \dots, n$, we also have

$$c \leq 3n + 2(n-1)(n_U + n_L + n_U 2 + n_L 2).$$

Thus, for values of n_L and n_U much less than n , the quasiseparability is useful as the multiplication can be carried out in $\mathcal{O}(n)$ arithmetic operations.

Note additionally that the implementation of the algorithm suggested by the above decomposition coincides with the algorithm derived differently in [11] for the same purpose.

5.4. Special choices of generators. The new Björck–Pereyra algorithm is based on the decomposition (3.4) and the fast matrix-vector product algorithm of section 5.3. The input of the latter algorithm is a generator of the corresponding (H, m) -quasiseparable matrix (section 5.2). However, in many examples, the generators of this form are not given explicitly, but rather by the recurrence relations of the corresponding (H, m) -quasiseparable polynomials.

We next provide a detailed conversion between two different representations, i.e., Table 5.1 lists formulas to compute the generators from the recurrence relations coefficients for all special cases considered above.

To illustrate a set of generators for a higher order matrix, the next example considers the m -recurrent polynomials of section 2.3.3.

Example 5.3. Consider the system of polynomials $\{r_0(x), \dots, r_5(x)\}$ that are 4-recurrent; that is, they satisfy (2.5) with $m = 4$:

$$r_k(x) = (\alpha_k x - a_{k-1,k}) \cdot r_{k-1}(x) - a_{k-2,k} \cdot r_{k-2}(x) - a_{k-3,k} \cdot r_{k-3}(x).$$

The corresponding confederate matrix is the following 5×5 matrix of the form in (2.6)

TABLE 5.1
Specific choices of generators resulting in various special cases.

Polynomials	p_k	q_k	d_k	g_k	b_k
Monomials (4.1)	1	1	0	0	0
Real orth. (4.4)	1	$1/\alpha_k$	δ_k/α_k	γ_k/α_k	0
Szegö (4.6)	1	μ_k	$-\rho_k \rho_{k-1}^*$	ρ_{k-1}^*	μ_{k-1}
Gen. three-term (2.7)	1	$1/\alpha_k$	$\frac{\delta_k}{\alpha_k} + \frac{\beta_k}{\alpha_{k-1}\alpha_k}$	$\frac{d_k\beta_{k+1} + \gamma_{k+1}}{\alpha_{k+1}}$	$\frac{\beta_{k+1}}{\alpha_{k+1}}$
Szegö-type (2.9)	1	$1/\delta_k$	$-\frac{\theta_k + \gamma_k \beta_{k-1}}{\delta_k}$	β_{k-1}	$\alpha_{k-1} - \beta_{k-1}\gamma_{k-1}$
EGO-type (2.11)	1	$1/\delta_k$	$-\frac{\theta_k}{\delta_k}$	β_k	α_k

Polynomials	h_k
Monomials (4.1)	1
Real orth. (4.4)	1
Szegö (4.6)	$-\mu_{k-1}\rho_k$
Gen. three-term (2.7)	1
Szegö-type (2.9)	$-\frac{\gamma_k}{\delta_k}(\alpha_{k-1} - \beta_{k-1}\gamma_{k-1})$
EGO-type (2.11)	$-\frac{\gamma_k}{\delta_k}$

with $m = 3$:

$$A = \begin{bmatrix} \frac{a_{0,1}}{\alpha_1} & \frac{a_{0,2}}{\alpha_2} & \frac{a_{0,3}}{\alpha_3} & 0 & 0 \\ \frac{a_{1,1}}{\alpha_1} & \frac{a_{1,2}}{\alpha_2} & \frac{a_{1,3}}{\alpha_3} & \frac{a_{1,4}}{\alpha_4} & 0 \\ 0 & \frac{1}{\alpha_2} & \frac{\alpha_3}{a_{2,3}} & \frac{\alpha_4}{a_{2,4}} & \frac{a_{2,5}}{\alpha_5} \\ 0 & 0 & \frac{\alpha_3}{1} & \frac{\alpha_4}{a_{3,4}} & \frac{\alpha_5}{a_{3,5}} \\ 0 & 0 & 0 & \frac{\alpha_4}{1} & \frac{\alpha_5}{a_{4,5}} \end{bmatrix}.$$

It can readily be seen that the matrices given by

$$g_1 = [a_{0,2} \ a_{0,3}], \quad g_2 = [a_{1,4} \ a_{1,3}], \quad g_3 = [a_{2,4} \ a_{2,5}], \quad g_4 = [0 \ a_{3,5}]$$

$$b_2 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad b_3 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad b_4 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

$$h_2 = \begin{bmatrix} \frac{1}{\alpha_2} \\ 0 \end{bmatrix}, \quad h_3 = \begin{bmatrix} 0 \\ \frac{1}{\alpha_3} \end{bmatrix}, \quad h_4 = \begin{bmatrix} \frac{1}{\alpha_4} \\ 0 \end{bmatrix}, \quad h_5 = \begin{bmatrix} 0 \\ \frac{1}{\alpha_5} \end{bmatrix},$$

are generators of the matrix A .

6. Derivation of the new Björck–Pereyra-type algorithm. In this section the algorithm presented is derived and the main enabling lemma is proved. The section begins with some background material.

6.1. Associated (generalized Horner) polynomials. Following [22] define the associated polynomials $\widehat{R} = \{\widehat{r}_0(x), \dots, \widehat{r}_n(x)\}$ for a given system of polynomials $R = \{r_0(x), \dots, r_n(x)\}$ via the relation

$$(6.1) \quad \frac{r_n(x) - r_n(y)}{x - y} = [r_0(x) \ r_1(x) \ r_2(x) \ \cdots \ r_{n-1}(x)] \cdot \begin{bmatrix} \widehat{r}_{n-1}(y) \\ \widehat{r}_{n-2}(y) \\ \vdots \\ \widehat{r}_1(y) \\ \widehat{r}_0(y) \end{bmatrix},$$

with additionally $\hat{r}_n(x) = r_n(x)$.

However, before proceeding we first clarify the existence of such polynomials. First, the polynomials associated with the monomials exist. Indeed, if P is the system of $n + 1$ polynomials $P = \{1, x, \dots, x^{n-1}, r_n(x)\}$, then

(6.2)

$$\frac{r_n(x) - r_n(y)}{x - y} = \begin{bmatrix} 1 & x & x^2 & \cdots & x^{n-1} \end{bmatrix} \cdot \begin{bmatrix} \hat{p}_{n-1}(y) \\ \hat{p}_{n-2}(y) \\ \vdots \\ \hat{p}_1(y) \\ \hat{p}_0(y) \end{bmatrix} = \sum_{i=0}^{n-1} x^i \cdot \hat{p}_{n-1-i}(y),$$

and in this case the associated polynomials \hat{P} can be seen to be the classical Horner polynomials (see, e.g., [22, section 3]).

Second, given a system of polynomials $R = \{r_0(x), r_1(x), \dots, r_{n-1}(x), r_n(x)\}$, there is a corresponding system of polynomials $\hat{R} = \{\hat{r}_0(x), \hat{r}_1(x), \dots, \hat{r}_{n-1}(x), \hat{r}_n(x)\}$ (with $\hat{r}_n(x) = r_n(x)$) satisfying (6.1). One can see that, given a polynomial system R with $\deg(r_k) = k$, the polynomials in R can be obtained from the monomial basis by

$$(6.3) \quad \begin{bmatrix} 1 & x & x^2 & \cdots & x^{n-1} \end{bmatrix} S = \begin{bmatrix} r_0(x) & r_1(x) & r_2(x) & \cdots & r_{n-1}(x) \end{bmatrix},$$

where S is an $n \times n$ upper triangular invertible matrix capturing the recurrence relations of the polynomial system R . Inserting SS^{-1} into (6.2) between the row and column vectors and using (6.3), we see that the polynomials associated with R are

$$(6.4) \quad \begin{bmatrix} \hat{r}_{n-1}(y) \\ \hat{r}_{n-2}(y) \\ \vdots \\ \hat{r}_1(y) \\ \hat{r}_0(y) \end{bmatrix} = S^{-1} \begin{bmatrix} \hat{p}_{n-1}(y) \\ \hat{p}_{n-2}(y) \\ \vdots \\ \hat{p}_1(y) \\ \hat{p}_0(y) \end{bmatrix},$$

where $\hat{P} = \{\hat{p}_0(x), \dots, \hat{p}_{n-1}(x)\}$ are the classical Horner polynomials and S is from (6.3).

The following lemma will be needed in the proof presented below.

LEMMA 6.1. *Let $R = \{r_0(x), \dots, r_{n-1}(x)\}$ be a system of polynomials satisfying (2.1), and for $k = 1, 2, \dots, n - 1$ denote by $R^{(k)}$ the system of polynomials $R^{(k)} = \{\hat{r}_0^{(k)}(x), \dots, \hat{r}_k^{(k)}(x)\}$ associated with the truncated system $\{r_0(x), \dots, r_k(x)\}$. Then*

$$(6.5) \quad \begin{bmatrix} \hat{r}_0^{(1)}(x) & \hat{r}_1^{(2)}(x) & \cdots & \hat{r}_{n-2}^{(n-1)}(x) \\ & \hat{r}_0^{(2)}(x) & \cdots & \hat{r}_{n-3}^{(n-1)}(x) \\ & & \ddots & \vdots \\ & & & \hat{r}_0^{(n-1)}(x) \end{bmatrix}^{-1} = \begin{bmatrix} \frac{1}{\alpha_1} & -x + \frac{\alpha_{1,2}}{\alpha_2} & \cdots & \frac{1}{\alpha_{n-1}} a_{1,n-1} \\ & \frac{1}{\alpha_2} & \cdots & \frac{1}{\alpha_{n-1}} a_{2,n-1} \\ & & \ddots & \vdots \\ & & & -x + \frac{a_{n-2,n-1}}{\alpha_{n-1}} \\ & & & \frac{1}{\alpha_{n-1}} \end{bmatrix}.$$

Proof. From [22] we have the formula

$$(6.6) \quad C_{\hat{R}}(\hat{r}_n) = \tilde{I} \cdot C_R(r_n)^T \cdot \tilde{I} \quad (\text{with } \hat{r}_n(x) = r_n(x)),$$

where \tilde{T} is the antidiagonal matrix, which provides a relation between the confederate matrix of a polynomial system R and that of the polynomials associated with R . From this we have the following n -term recurrence relations for the truncated associated polynomials:

$$(6.7) \quad \hat{r}_m^{(k)}(x) = \alpha_m \left[\left(x - \frac{a_{m,m+1}}{\alpha_{m+1}} \right) \hat{r}_{m-1}^{(k)} - \frac{a_{m,m+2}}{\alpha_{m+2}} \hat{r}_{m-2}^{(k)} - \dots - \frac{a_{m,k}}{\alpha_k} \hat{r}_0^{(k)} \right], \\ m = 1, \dots, k-1,$$

with

$$(6.8) \quad \hat{r}_0^{(k)} = 1/\alpha_k.$$

Now consider the product

$$(6.9) \quad \begin{bmatrix} \frac{1}{\alpha_1} & -x + \frac{a_{1,2}}{\alpha_2} & \cdots & \cdots & \frac{a_{1,n-1}}{\alpha_{n-1}} \\ & \ddots & \ddots & & \vdots \\ & & \frac{1}{\alpha_i} & -x + \frac{a_{i,i+1}}{\alpha_{i+1}} & \cdots & \frac{a_{i,n-1}}{\alpha_{n-1}} \\ & & & \ddots & -x + \frac{a_{n-2,n-1}}{\alpha_{n-1}} \\ & & & & \frac{1}{\alpha_{n-1}} \end{bmatrix} \cdot \begin{bmatrix} \hat{r}_0^{(1)}(x) & \cdots & \hat{r}_{j-1}^{(j)}(x) & \cdots & \hat{r}_{n-2}^{(n-1)}(x) \\ & & \hat{r}_{j-2}^{(j)}(x) & & \hat{r}_{n-3}^{(n-1)}(x) \\ & & \vdots & & \vdots \\ & & \hat{r}_0^{(j)}(x) & & \hat{r}_1^{(n-1)}(x) \\ & & & \ddots & \hat{r}_0^{(n-1)}(x) \end{bmatrix}.$$

The (i, j) entry of this product defined by the highlighted row and column can be seen as (6.7) with $k = j, m = j - i$ if $i \neq j$ and (6.8) with $k = i, m = 0$ if $i = j$. Thus this product is the identity, implying (6.5). \square

With this completed, the proof of Lemma 3.1 from section 3 is next.

Proof. Performing one step of Gaussian elimination on $V_R(x_{1:n})$ yields

$$(6.10) \quad V_R(x_{1:n}) = \begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ \vdots & & \ddots & \\ 1 & & & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & & & \\ & x_2 - x_1 & & \\ & & \ddots & \\ & & & x_n - x_1 \end{bmatrix} \\ \cdot \begin{bmatrix} 1 & 0 \\ 0 & \bar{R} \end{bmatrix} \cdot \left[\begin{array}{c|ccc} r_0(x_1) & r_1(x_1) & \cdots & r_{n-1}(x_1) \\ \hline 0 & & & I \end{array} \right],$$

where the matrix \bar{R} has (i, m) -entry $\bar{R}_{i,m} = \frac{r_{m+1}(x_{i+1}) - r_1(x_1)}{x_{i+1} - x_1}$; that is, \bar{R} consists of divided differences. By the discussion above, associated with the system R is the system $\hat{R} = \{\hat{r}_0(x), \dots, \hat{r}_n(x)\}$. Following the notation of Lemma 6.1, denote by $\hat{R}^{(k)} = \{\hat{r}_0^{(k)}(x), \dots, \hat{r}_k^{(k)}(x)\}$ the system of polynomials associated with the truncated system $\{r_0(x), \dots, r_k(x)\}$. By the definition of the associated polynomials we have

for $k = 1, 2, \dots, n-1$

$$\frac{r_k(x) - r_k(y)}{x - y} = \begin{bmatrix} r_0(x) & r_1(x) & r_2(x) & \cdots & r_{k-1}(x) \end{bmatrix} \cdot \begin{bmatrix} \hat{r}_{k-1}^{(k)}(y) \\ \hat{r}_{n-2}^{(k)}(y) \\ \vdots \\ \hat{r}_1^{(k)}(y) \\ \hat{r}_0^{(k)}(y) \end{bmatrix}$$

$$= \sum_{i=0}^{k-1} r_i(x) \cdot \hat{r}_{k-1-i}^{(k)}(y).$$

Finally, denoting by $\widehat{R}^{(k)} = \{\hat{r}_0^{(k)}(x), \dots, \hat{r}_k^{(k)}(x)\}$ the system of polynomials associated with the truncated system $\{r_0(x), \dots, r_k(x)\}$ we can further factor \bar{R} as

$$(6.11) \quad \bar{R} = V_R(x_{1:n}) \cdot \begin{bmatrix} \hat{r}_0^{(1)}(x_j) & \hat{r}_1^{(2)}(x_j) & \cdots & \hat{r}_{n-2}^{(n-1)}(x_j) \\ & \hat{r}_0^{(2)}(x_j) & \cdots & \hat{r}_{n-3}^{(n-1)}(x_j) \\ & & \ddots & \vdots \\ & & & \hat{r}_0^{(n-1)}(x_j) \end{bmatrix}.$$

The last matrix on the right-hand side of (6.11) can be inverted by Lemma 6.1. Therefore, inverting (6.10) and substituting (6.5) results in (3.1) as desired. \square

7. Numerical illustrations. We report here several results of our preliminary numerical experiments to indicate that in the generic case the behavior of the generalized algorithms is consistent with the conclusions reported earlier for the known Björck–Pereyra-type algorithms. All experiments are given for the $(H, 1)$ -quasiseparable case.

The algorithm has been implemented in MATLAB version 7 and, using MATLAB-implemented single and double precision, we compare the solutions obtained in each of these cases. That is, we use the solution obtained from the algorithm in double precision as the exact solution, and compare it with the solution obtained in single precision.

It is known (see [20, 30]) that reordering the nodes for polynomial Vandermonde matrices, which corresponds to a permutation of the rows, can affect the accuracy of related algorithms. In particular, ordering the nodes according to the *Leja ordering*

$$|x_1| = \max_{1 \leq i \leq n} |x_i|, \quad \prod_{j=1}^{k-1} |x_k - x_j| = \max_{k \leq i \leq n} \prod_{j=1}^{k-1} |x_i - x_j|, \quad k = 2, \dots, n-1,$$

(see [20, 27, 30]) improves the performance of many similar algorithms. We include experiments with and without the use of Leja ordering (if the Leja ordering is not used, the nodes are randomly ordered). A counterpart of this ordering is known for Cauchy matrices; see [5].

In all experiments, we compare the relative forward accuracy of the algorithm, defined by

$$e = \frac{\|x - \hat{x}\|_2}{\|x\|_2},$$

TABLE 7.1
Equidistant nodes on $(-1, 1)$.

n	$\text{cond}(V)$	$\gamma(A, f)$	GEPP	BP-QS	BP-QS-L
10	1e+009	3e+000	1e-004	2e-004	2e-006
10	2e+011	2e+000	2e-005	7e-007	2e-007
10	2e+005	2e+000	3e-005	2e-006	8e-007
15	7e+011	1e+001	5e-003	5e-005	2e-006
15	7e+012	4e+001	2e-004	7e-004	6e-007
15	9e+013	1e+001	4e-001	3e-005	2e-007
20	4e+018	4e+000	1e-001	5e-004	7e-007
20	7e+015	7e+000	9e-002	3e-003	5e-007
20	6e+014	3e+000	4e-001	4e-003	7e-007
25	2e+017	1e+001	9e-001	1e-001	1e-006
25	2e+020	5e+000	2e+000	2e+000	1e-006
25	5e+024	8e+000	2e+001	3e-002	1e-006
30	3e+025	8e+000	1e+000	6e+000	1e-006
30	1e+020	4e+000	1e+000	4e+000	1e-006
30	4e+026	4e+000	1e+000	1e+000	1e-006
35	5e+023	4e+001	1e+000	3e+002	3e-006
35	1e+027	1e+001	1e+000	3e+007	6e-006
35	3e+035	8e+000	1e+000	2e+002	6e-007
40	5e+031	6e+000	1e+000	1e+003	5e-007
40	6e+035	4e+001	1e+000	9e+002	6e-008
40	3e+037	7e+001	1e+000	3e+003	2e-007
45	6e+031	4e+001	1e+000	1e+006	1e-006
45	1e+035	7e+000	1e+000	1e+009	7e-006
45	7e+032	1e+001	1e+000	6e+004	2e-006
50	8e+029	1e+001	1e+000	5e+006	5e-007
50	6e+035	9e+001	1e+000	7e+010	2e-006
50	1e+036	3e+001	1e+000	1e+007	2e-007

where \hat{x} is the solution computed by each algorithm in MATLAB in single precision, and x is the exact solution computed in double precision. In Tables 7.1–7.4, BP-QS denotes the proposed Björck–Pereyra like algorithm with a random ordering of the nodes, and BP-QS-L denotes the same algorithm using the Leja ordering. The factors L_k from (3.3) were used. GEPP indicates MATLAB’s Gaussian elimination. Finally, $\text{cond}(V)$ denotes the condition number of the matrix V with respect to $\|\cdot\|_2$, computed via the MATLAB command `cond()`.

We next briefly remind the reader of the concept of *effective well-conditioning*. In [8], Chan and Foulser introduced this concept, motivated by the high relative accuracy of the BP algorithm for sign-oscillating right-hand side, and by some other examples. The latter reflects the fact that the sensitivity to perturbations of a solution depends upon the direction of the right-hand side vector, which suggests the potential existence of algorithms that can exploit the effective well-conditioning to produce higher relative accuracy for special right-hand sides. Note, however, that in general the numerical performance of a certain algorithm and the effective well-conditioning of a system to be solved have nothing to do with each other, and many algorithms are insensitive to the direction of the right-hand side vector.

Here we use this approach to show that the proposed Björck–Pereyra-like algorithm is indeed an example of an algorithm that accurately solves effectively well-conditioned polynomial Vandermonde systems. Before stating the result, let us introduce the necessary notations.

Recall that $\theta(A, f)$ measures, in the infinity-norm, the sensitivity of the solution of $Aa = f$ to small componentwise bounded perturbations in the right-hand side. The

TABLE 7.2
Clustered nodes on $(-1, 1)$.

n	cond(V)	$\gamma(A, f)$	GEPP	BP-QS	BP-QS-L
10	3e+011	2e+000	2e-005	2e-005	1e-008
10	2e+010	6e+000	5e-005	1e-005	4e-007
10	7e+008	2e+000	6e-006	2e-005	5e-007
15	4e+014	2e+000	1e-001	1e-003	9e-008
15	1e+011	4e+000	2e-002	2e-004	2e-007
15	5e+009	3e+000	3e-002	2e-005	3e-007
20	6e+015	3e+000	1e+000	3e-003	2e-007
20	5e+017	4e+000	8e+000	6e-002	1e-005
20	3e+013	2e+000	1e+000	5e-003	4e-007
25	2e+025	4e+000	1e+000	3e-003	7e-007
25	1e+017	2e+000	1e+000	7e-001	1e-006
25	4e+013	3e+000	1e+000	1e-001	1e-006
30	1e+023	8e+000	1e+000	3e+000	2e-006
30	2e+025	7e+000	1e+000	6e-001	7e-006
30	4e+029	1e+001	1e+000	7e+001	9e-005
35	1e+029	6e+000	1e+000	8e+001	6e-007
35	6e+026	1e+001	1e+000	5e+001	2e-006
35	3e+031	3e+000	1e+000	6e+001	2e-007
40	2e+035	2e+002	1e+000	2e+003	1e-006
40	8e+035	2e+001	1e+000	1e+007	2e-005
40	6e+027	7e+000	1e+000	1e+006	2e-006
45	3e+033	7e+000	1e+000	1e+008	3e-007
45	4e+034	7e+000	1e+000	2e+008	4e-005
45	4e+035	6e+001	1e+000	1e+010	1e-006
50	2e+036	8e+000	1e+000	3e+006	4e-007
50	7e+036	7e+001	1e+000	2e+009	5e-006
50	8e+035	3e+001	1e+000	1e+009	2e-006

2-norm condition number associated with the same map is given by

$$(7.1) \quad \kappa_2(A, f) = \lim_{\varepsilon \rightarrow 0} \sup_{\|\Delta f\|_2 \leq \varepsilon \|f\|_2} \frac{\|\Delta a\|_2}{\|a\|_2 \cdot \varepsilon},$$

i.e., it assumes that both the perturbation in f and the variation in a are measured in the 2-norm. Chan and Foulser derived an upper bound for $\kappa_2(A, f)$ in terms of the direction of the right-hand side vector relative to the left singular vectors of A . To be more precise, let $A = U \cdot \Sigma \cdot V^T$ be the singular value decomposition, where the columns of $U = [u_1 \ u_2 \ \cdots \ u_n]$ are the left singular vectors, the columns of $V = [v_1 \ v_2 \ \cdots \ v_n]$ are the right singular vectors, and the diagonal entries of $\Sigma \text{diag}\{\sigma_1 \ \sigma_2 \ \cdots \ \sigma_n\}$ are the singular values of A in decreasing order. Denote by

$$P_k = [u_{n+1-k} \ \cdots \ u_n] \cdot [u_{n+1-k} \ \cdots \ u_n]^T$$

the projection operator onto the linear span of the smallest k left singular vectors of A . Then, in accordance with [8],

$$(7.2) \quad \kappa_2(A, f) \leq \gamma(A, f) \leq \kappa_2(A),$$

where

$$(7.3) \quad \gamma(A, f) = \min_k \frac{\sigma_{n-k+1}}{\sigma_n} \cdot \frac{\|f\|_2}{\|P_k f\|_2},$$

TABLE 7.3
Banded matrices, various bandwidths.

n	Bands	cond(V)	$\gamma(A, f)$	GEPP	BP-QS	BP-QS-L
10	1	7e+007	5e+000	6e-005	4e-006	2e-007
10	2	6e+007	1e+001	6e-005	3e-006	2e-007
10	3	9e+006	5e+000	1e-006	2e-007	2e-007
10	4	3e+007	3e+001	9e-005	3e-006	2e-006
10	5	1e+008	8e+000	2e-004	9e-006	5e-007
15	1	3e+012	3e+000	1e-002	6e-005	1e-007
15	2	1e+014	5e+000	2e-001	1e-004	2e-006
15	3	2e+009	4e+000	1e-004	2e-004	1e-006
15	4	3e+013	2e+001	7e-002	3e-004	1e-006
15	5	6e+013	7e+000	5e-003	5e-005	7e-007
20	1	2e+015	1e+001	9e+000	3e-003	2e-007
20	2	2e+010	5e+000	8e-002	2e-003	8e-007
20	3	1e+020	2e+001	1e+000	2e-002	2e-007
20	4	6e+016	8e+000	2e-002	5e-003	1e-007
20	5	6e+015	1e+001	1e+000	1e-002	3e-007
25	1	1e+024	5e+000	2e+000	3e-001	3e-007
25	2	1e+025	5e+000	3e+001	1e+000	6e-007
25	3	5e+017	4e+000	2e+000	6e-003	1e-006
25	4	2e+016	4e+001	6e-002	2e+000	1e-006
25	5	5e+019	7e+003	2e+000	2e-002	3e-007
30	1	1e+028	3e+000	1e+000	1e+000	1e-006
30	2	3e+024	4e+000	1e+000	4e+000	5e-007
30	3	2e+033	2e+001	1e+000	5e+000	5e-006
30	4	5e+024	1e+001	1e+000	1e+001	1e-006
30	5	4e+019	2e+000	9e-001	6e+001	1e-005

and $\kappa_2(A) = \|A^{-1}\|_2 \cdot \|A\|_2$ is the 2-norm condition number. Note that the second inequality in (7.2) can easily be deduced by inspecting (7.3) for $k = n$. At the same time, if a large part of f lies in the span of the small left singular vectors of A , then $\gamma(A, f)$ can be much smaller than $\kappa_2(A)$, thus providing a better bound for $\kappa_2(A, f)$ in (7.2). Linear systems for which the *Chan–Foulser number* in (7.3) is much smaller than $\kappa_2(A)$ are called *effectively well-conditioned*. We shall refer to an algorithm as *effectively (forward) stable*, if it is guaranteed to produce high relative accuracy for effectively well-conditioned systems. Backward effective stability can be defined in the same way.

We include in all tables that the Chan–Foulser condition number $\gamma(A, f)$ is presented for comparison with the standard condition number and associated results.

EXPERIMENT 1. In Table 7.1, the values for the generators were chosen randomly on $(-1, 1)$, similarly for the entries of the right-hand side vector. The nodes x_k were selected equidistant on $(-1, 1)$ via the formula

$$x_k = -1 + 2 \left(\frac{k}{n-1} \right), \quad k = 0, 1, \dots, n-1.$$

We test the relative accuracy of the algorithm for various sizes n of matrices generated in this way.

Notice that as the size of the matrices involved rises, so does the condition number of the matrices, and hence as expected, the performance of Gaussian elimination declines. The performance of the proposed algorithm with a random ordering of the nodes is not much of an improvement over that of GE; however, using the Leja

ordering gives a dramatic improvement in performance in this case.

EXPERIMENT 2. In Table 7.2, the values for the generators and entries of the right-hand side vector were chosen as in Experiment 1, and the nodes x_k were selected clustered on $(-1, 1)$ via the formula

$$x_k = -1 + 2 \left(\frac{k}{n-1} \right)^2, \quad k = 0, 1, \dots, n-1.$$

Again we test the relative accuracy for various $n \times n$ matrices generated in this way.

In this experiment, again the condition number rises with the size of the matrix, and as expected Gaussian elimination gives less relative accuracy as this increases. The proposed algorithm gives an improvement in this case as well, and the Leja ordering again gives an improvement.

EXPERIMENT 3. In the third experiment, we present a numerical example of the motivating special case of m -recurrent polynomials, corresponding to banded matrices. The generators were chosen in such a way as to produce the desired banded structure (see section 5.4), and those entries that were chosen nonzero were chosen randomly on $(-1, 1)$, as were the entries of the right-hand side vector. As in Experiment 1, the nodes were chosen equidistant on $[-1, 1]$.

For each size n , different bandwidths were considered from one (meaning only one nonzero diagonal above the main diagonal, or tridiagonal) through five, or five nonzero diagonals above the main diagonal. The results are shown in Table 7.3.

EXPERIMENT 4. In the next experiment, we show that these experiments are consistent with observations in [4] and [8]. This is illustrated in Table 7.4, which shows the results for a 30×30 matrix with condition number $\text{cond}(V) = 2e + 021$ generated by a fixed set of generators and nodes randomly chosen on $(-1, 1)$, and the results of applying the various algorithms to solve the system with each (left) singular vector as the right-hand side.

The outcome is consistent with observations in [4] and [8] for Björck–Pereyra-type algorithms for the classical Vandermonde and Cauchy matrices.

The above typical experiments describe generic Hessenberg-quasiseparable cases, and they are preliminary. We conclude this section with two comments about further possible work. First, experience indicates that numerical properties of general polynomial algorithms may be quite different even for two classical special cases: real orthogonal polynomials and Szegő polynomials. Hence it is of interest to study the numerical behavior of the new algorithm for different special cases, e.g., four examples described in sections 2.3.3–2.3.6.

Second, there is an ongoing work [6] on studying the accuracy of fast algorithms for multiplying a quasiseparable matrix by a vector. There are several alternatives to the method described above in section 5.3, and one of them is provably stable. When the work [6] is completed, it will be of interest to study the accuracy of our algorithm based on (3.4) combined with new methods for computing quasiseparable matrix-vector product.

However, already at this point the results of all of our experiments are fully consistent with the conclusions made in [3, 4, 7, 8, 20, 27, 30], for similar algorithms. Specifically, there are examples, even in the most generic Hessenberg-quasiseparable case, in which the Björck–Pereyra-type algorithms can yield a very high relative forward accuracy.

TABLE 7.4
 Dependence on the direction using left singular vectors. $\text{cond}(V)=2e+021$.

Singular vector u_k	$\text{cond}(V)$	$\gamma(A, f)$	GEPP	BP-QS	BP-QS-L
1	1e+018	8e+008	1e+000	2e+004	1e+005
2	1e+018	6e+007	1e+000	1e+003	1e+006
3	1e+018	4e+006	1e+000	2e+001	3e+001
4	1e+018	5e+005	1e+000	4e+000	8e+000
5	1e+018	3e+005	1e+000	2e+001	3e+001
6	1e+018	4e+004	1e+000	2e+001	4e+000
7	1e+018	5e+004	1e+000	2e+001	5e+000
8	1e+018	1e+006	1e+000	1e+000	2e-001
9	1e+018	5e+005	1e+000	2e-001	8e+000
10	1e+018	4e+004	1e+000	6e-001	4e-001
11	1e+018	7e+003	1e+000	8e-001	5e-001
12	1e+018	5e+003	1e+000	4e+000	2e+000
13	1e+018	3e+003	1e+000	2e+001	2e+001
14	1e+018	1e+004	1e+000	2e-001	4e-001
15	1e+018	3e+003	1e+000	2e+000	2e+000
16	1e+018	4e+002	1e+000	7e-001	1e+000
17	1e+018	4e+002	1e+000	4e-001	2e+000
18	1e+018	6e+002	1e+000	3e+000	4e+000
19	1e+018	3e+002	1e+000	4e+000	2e+000
20	1e+018	2e+002	7e-001	6e+000	2e+001
21	1e+018	5e+001	1e+000	1e+000	3e-001
22	1e+018	2e+001	7e-001	2e+000	2e+000
23	1e+018	2e+001	1e+000	2e-002	7e-002
24	1e+018	1e+001	1e+000	9e-007	3e-007
25	1e+018	2e+000	1e+000	1e-007	4e-007
26	1e+018	2e+001	1e+000	4e-007	4e-007
27	1e+018	1e+001	1e+000	5e-007	3e-007
28	1e+018	3e+000	1e+000	3e-006	5e-006
29	1e+018	1e+000	1e+000	2e-007	3e-007
30	1e+018	5e+000	1e+000	6e-007	2e-007

8. Conclusion. In this paper we generalized the well-known Björck–Pereyra algorithm to polynomial-Vandermonde matrices. The efficiency of the algorithm depends on the structure of the corresponding confederate matrix (a Hessenberg matrix capturing the recurrence relations). In the case where the polynomial system is (H, m) -quasiseparable with small m (i.e., when the confederate matrix is quasiseparable) the algorithm has a favorably complexity of $\mathcal{O}(n^2)$ operations, which is an order of magnitude improvement over the standard methods. Initial numerical experiments indicate that in many cases the algorithm provides better forward accuracy than the one of Gaussian elimination. This observation is fully consistent with the experience our colleagues reported in several previous papers.

REFERENCES

- [1] M. BAKONYI AND T. CONSTANTINESCU, *Schur's Algorithm and Several Applications*, in Pitman Res. Notes in Math. Ser., vol. 61, Longman Scientific and Technical, Harlow, UK, 1992.
- [2] T. BELLA, Y. EIDELMAN, I. GOHBERG, V. OLSHEVSKY, AND E. TYRTYSHNIKOV, *Fast Inversion of Hessenberg-Quasiseparable Vandermonde Matrices*, submitted.
- [3] T. BELLA, Y. EIDELMAN, I. GOHBERG, I. KOLTRACHT, AND V. OLSHEVSKY, *A Björck-Pereyra-type algorithm for Szegő-Vandermonde matrices based on properties of unitary Hessenberg matrices*, Linear Algebra Appl., 420 (2007), pp. 634–647.
- [4] T. BOROS, T. KAILATH, AND V. OLSHEVSKY, *A fast parallel Björck-Pereyra-type algorithm for*

- solving Cauchy linear equations*, Linear Algebra Appl., 302/303 (1999), pp. 265–293.
- [5] T. BOROS, T. KAILATH, AND V. OLSHEVSKY, *Pivoting and backward stability of fast algorithms for solving Cauchy linear equations*, 343/344 (2002), pp. 63–99.
- [6] T. BELLA, V. OLSHEVSKY, AND M. STEWART, *Accuracy of Several Fast Algorithms to Multiply a Quasiseparable Matrix by a Vector*, in preparation.
- [7] A. BJÖRCK AND V. PEREYRA, *Solution of Vandermonde systems of equations*, Math. Comp., 24 (1970), pp. 893–903.
- [8] T. F. CHAN AND D. E. FOULSER, *Effectively well-conditioned linear systems*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 963–969.
- [9] J. DEMMEL AND P. KOEV, *The accurate and efficient solution of a totally positive generalized Vandermonde linear system*, SIAM J. Matrix Anal. Appl., 27 (2005), pp. 142–152.
- [10] Y. EIDELMAN AND I. GOHBERG, *On a new class of structured matrices*, Integral Equations Operator Theory, 34 (1999), pp. 293–324.
- [11] Y. EIDELMAN AND I. GOHBERG, *Linear complexity inversion algorithms for a class of structured matrices*, Integral Equations Operator Theory, 35 (1999), pp. 28–52.
- [12] Y. EIDELMAN AND I. GOHBERG, *A modification of the Dewilde-van der Veen method for inversion of finite structured matrices*, Linear Algebra Appl., 343/344 (2002), pp. 419–450.
- [13] Y. EIDELMAN, I. GOHBERG, AND V. OLSHEVSKY, *Eigenstructure of order-one-quasiseparable matrices. Three-term and two-term recurrence relations*, Linear Algebra Appl., 405 (2005), pp. 1–40.
- [14] W. B. GRAGG, *Positive definite Toeplitz matrices, the Arnoldi process for isometric operators, and Gaussian quadrature on the unit circle*, J. Comput. Appl. Math., 46 (1993), pp. 183–198.
- [15] L. Y. GERONIMUS, *Polynomials orthogonal on a circle and their applications*, in Amer. Math. Soc. Transl., Vol. 3, Providence, RI, 1954, pp. 1–78.
- [16] I. GOHBERG AND V. OLSHEVSKY, *Fast inversion of Chebyshev-Vandermonde matrices*, Numer. Math., 67 (1994), pp. 71–92.
- [17] U. GRENADER AND G. SZEGÖ, *Toeplitz Forms and Applications*, University of California Press, Berkeley, CA, 1958.
- [18] G. GOLUB AND C. VAN LOAN, *Matrix Computations*, 3rd ed., Johns Hopkins University Press, Baltimore, MD, 1996.
- [19] N. J. HIGHAM, *Error analysis of the Björck–Pereyra algorithms for solving Vandermonde systems*, Numer. Math., 50 (1987), pp. 613–632.
- [20] N. J. HIGHAM, *Stability analysis of algorithms for solving confluent Vandermonde-like systems*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 23–41.
- [21] N. J. HIGHAM, *Accuracy and Stability of Numerical Algorithms*, 2nd ed., SIAM, Philadelphia, 2002.
- [22] T. KAILATH AND V. OLSHEVSKY, *Displacement-structure approach to polynomial Vandermonde and related matrices*, Linear Algebra Appl., 261 (1997), pp. 49–90.
- [23] J. MAROULAS AND S. BARNETT, *Polynomials with respect to a general basis. I. Theory*, J. Math. Anal. Appl., 72 (1979), pp. 177–194.
- [24] A. MARCO AND J.-J. MARTINEZ, *A fast and accurate algorithm for solving Bernstein-Vandermonde linear systems*, Linear Algebra Appl., 422 (2007), pp. 616–628.
- [25] V. OLSHEVSKY, *Eigenvector computation for almost unitary Hessenberg matrices and inversion of Szegő-Vandermonde matrices via discrete transmission lines*, Linear Algebra Appl., 285 (1998), pp. 37–67.
- [26] V. OLSHEVSKY, *Associated polynomials, unitary Hessenberg matrices and fast generalized Parker-Traub and Björck-Pereyra algorithms for Szegő-Vandermonde matrices*, in Structured Matrices: Recent Developments in Theory and Computation, D. Bini, E. Tyrtyshnikov, and P. Yalamov, eds., 2001, NOVA Science Publ., pp. 67–78.
- [27] V. OLSHEVSKY, *Pivoting for structured matrices and rational tangential interpolation*, in Fast Algorithms for Structured Matrices: Theory and Applications, Contemp. Math. 323, Amer. Math. Soc., Providence, RI, 2003, pp. 1–73.
- [28] L. REICHEL, *Newton interpolation at Leja points*, BIT, 30 (1990), pp. 332–346.
- [29] P. A. REGALIA, *Adaptive IIR Filtering in Signal Processing and Control*, Marcel Dekker, New York, 1995.
- [30] L. REICHEL AND G. OFFER, *Chebyshev-Vandermonde systems*, Math. Comp., 57 (1991), pp. 703–721.
- [31] E. E. TYRTYSHNIKOV, *How bad are Hankel matrices?* Numer. Math., 67 (1994) pp. 261–269.
- [32] W. P. TANG AND G. H. GOLUB, *The block decomposition of a Vandermonde matrix and its applications*, BIT, 21 (1981), pp. 505–517.