



A Björck–Pereyra-type algorithm for Szegő–Vandermonde matrices based on properties of unitary Hessenberg matrices [☆]

Tom Bella ^{a,*}, Yuli Eidelman ^b, Israel Gohberg ^b,
Israel Koltracht ^a, Vadim Olshevsky ^a

^a Department of Mathematics, University of Connecticut, 196 Auditorium Road, Storrs, CT 06269-3009, USA

^b School of Mathematical Sciences, Raymond and Beverly Sackler Faculty of Exact Sciences,
Tel Aviv University, Ramat-Aviv 69978, Israel

Received 3 December 2005; accepted 24 August 2006

Available online 3 November 2006

Submitted by M. Tsatsomeros

Abstract

In this paper we carry over the Björck–Pereyra algorithm for solving Vandermonde linear systems to what we suggest to call Szegő–Vandermonde systems $V_{\Phi}(x)$, i.e., polynomial-Vandermonde systems where the corresponding polynomial system Φ is the *Szegő polynomials*. The properties of the corresponding *unitary Hessenberg* matrix allow us to derive a fast $O(n^2)$ computational procedure. We present numerical experiments that indicate that for ill-conditioned matrices the new algorithm yields better forward accuracy than Gaussian elimination.

© 2006 Elsevier Inc. All rights reserved.

AMS classification: 15A06; 65F05

Keywords: Björck–Pereyra algorithm; Szegő polynomials; Unitary Hessenberg matrices; Vandermonde matrices; Polynomial-Vandermonde matrices; Fast algorithms

[☆] Supported by the NSF grant 0242518.

* Corresponding author.

E-mail addresses: bella@math.uconn.edu (T. Bella), eideyu@post.tau.ac.il (Y. Eidelman), gohberg@post.tau.ac.il (I. Gohberg), koltracht@math.uconn.edu (I. Koltracht), olshevsky@math.uconn.edu (V. Olshevsky).

1. Introduction

Vandermonde matrices of the form

$$V(x) = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^{n-1} \end{bmatrix} \tag{1.1}$$

are classical, and explicit expressions for their determinants and inverses are well known. The structure in (1.1) can be exploited to speed-up computations involving $V(x)$ allowing one to design *fast algorithms*, i.e., algorithms whose complexity is at least an order of magnitude less than that of standard (structure-ignoring) methods.

For example, solving a Vandermonde linear system using methods that ignore this special structure requires $O(n^3)$ operations, whereas the now well known Björck–Pereyra algorithm (see [6,18,27]) solves the system in $O(n^2)$ operations. Moreover, it was shown that the Björck–Pereyra algorithm is not only faster but it is often more accurate than the standard methods, see, e.g. [19] for the forward stability and [4] for the backward stability analyses. This is despite the fact that the matrices involved are extremely ill-conditioned (see [33]).

Classical Vandermonde matrices (1.1) appear in polynomial computations exploiting the monomial basis $\{1, x, x^2, \dots, x^{n-1}\}$. A slightly more general class of matrices arise by following the same essential structure in (1.1), but permitting polynomials in place of the monomials. Such matrices are polynomial-Vandermonde matrices of the form

$$V_R(x) = \begin{bmatrix} r_0(x_1) & r_1(x_1) & \cdots & r_{n-1}(x_1) \\ r_0(x_2) & r_1(x_2) & \cdots & r_{n-1}(x_2) \\ \vdots & \vdots & & \vdots \\ r_0(x_n) & r_1(x_n) & \cdots & r_{n-1}(x_n) \end{bmatrix}, \tag{1.2}$$

where $R = \{r_0(x), r_1(x), \dots, r_{n-1}(x)\}$ is a given system of polynomials. The Björck–Pereyra algorithm for solving linear systems as well as the Traub algorithm for inversion have been extended to polynomial-Vandermonde matrices in several notable special cases of the polynomial system R . The resulting fast algorithms along with corresponding references are listed in Table 1.

However, many problems involve computations with the Szegő polynomials $\Phi = \{\phi_k^\#(x)\}$; that is, polynomials orthonormal on the unit circle with respect to a suitable inner product,

$$\langle p(x), q(x) \rangle = \frac{1}{2\pi} \int_{-\pi}^{\pi} p(e^{i\theta}) \cdot [q(e^{i\theta})]^* w^2(\theta) d\theta. \tag{1.3}$$

It is well known that the Szegő polynomials are completely described by the two-term recurrence relations

Table 1
Fast $O(n^2)$ algorithms for three-term Vandermonde matrices

Coefficient matrix	Inversion algorithm	Algorithm solving linear system
Vandermonde	Parker–Forney–Traub algorithm [28,10,32]	Björck–Pereyra algorithm [6]
Chebyshev–Vandermonde	Gohberg–Olshevsky algorithm [15]	Reichel–Opfer algorithm [31]
Three-term Vandermonde	Calvetti–Reichel algorithm [9]	Higham algorithm [20]
Szegő–Vandermonde	Olshevsky algorithm [26]	???

$$\begin{aligned} \begin{bmatrix} \phi_0(x) \\ \phi_0^\#(x) \end{bmatrix} &= \frac{1}{\mu_0} \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \\ \begin{bmatrix} \phi_{k+1}(x) \\ \phi_{k+1}^\#(x) \end{bmatrix} &= \frac{1}{\mu_{k+1}} \begin{bmatrix} 1 & -\rho_{k+1}^* \\ -\rho_{k+1} & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & x \end{bmatrix} \begin{bmatrix} \phi_k(x) \\ \phi_k^\#(x) \end{bmatrix}, \end{aligned} \tag{1.4}$$

see [17,14]. The parameters $\{\rho_0, \rho_1, \dots, \rho_n\}$ (with $\rho_0 := -1$), are called *reflection coefficients* (the names *parcor coefficients* and *Schur parameters* are also in use). The numbers $\mu_k = \sqrt{1 - |\rho_k|^2}$ are called the *complementary parameters* ($\mu_k := 1$ if $|\rho_k| = 1$), and $\phi_k(x) = x^k [\phi^\#(\frac{1}{x^*})]^*$.

We use the notation $\phi_k^\#(x)$ for the Szegő polynomials following [25,26] and the engineering references therein where $\phi_k(x)$ are called *backward predictor polynomials*, and the Szegő polynomials $\phi_k^\#(x)$ are obtained from them.

In this paper we present a Björck–Pereyra-type algorithm to solve linear systems where the coefficient matrix is a polynomial-Vandermonde matrix whose corresponding system of polynomials are Szegő polynomials. We call such a matrix a Szegő–Vandermonde matrix. This algorithm will be based on the Hessenberg matrix

$$H = \begin{bmatrix} -\rho_1 \rho_0^* & -\rho_2 \mu_1 \rho_0^* & -\rho_3 \mu_2 \mu_1 \rho_0^* & \cdots & -\rho_{n-1} \mu_{n-2} \cdots \mu_1 \rho_0^* & -\rho_n \mu_{n-1} \cdots \mu_1 \rho_0^* \\ \mu_1 & -\rho_2 \rho_1^* & -\rho_3 \mu_2 \rho_1^* & \cdots & -\rho_{n-1} \mu_{n-2} \cdots \mu_2 \rho_1^* & -\rho_n \mu_{n-1} \cdots \mu_2 \rho_1^* \\ 0 & \mu_2 & -\rho_3 \rho_2^* & \cdots & -\rho_{n-1} \mu_{n-2} \cdots \mu_3 \rho_2^* & -\rho_n \mu_{n-1} \cdots \mu_3 \rho_2^* \\ \vdots & \ddots & \mu_3 & & \vdots & \vdots \\ \vdots & & \ddots & \ddots & -\rho_{n-1} \rho_{n-2}^* & -\rho_n \mu_{n-1} \rho_{n-2}^* \\ 0 & \cdots & \cdots & 0 & \mu_{n-1} & -\rho_n \rho_{n-1}^* \end{bmatrix} \tag{1.5}$$

that was used in many areas, see, e.g., [12,3,30], and the references therein. This matrix has many nice properties. For instance, it is well known that H differs from a unitary matrix only in the last column. Additionally, it can be seen that if H_k is the leading principal $k \times k$ submatrix of H , then

$$\det(xI - H_k) = \frac{1}{\mu_1 \cdots \mu_k} \phi_k^\#(x) \tag{1.6}$$

see, for instance, [12].

Such almost-unitary Hessenberg matrices have been studied in many contexts, notably in the fields of numerical linear algebra, operator theory, and signal processing. In numerical linear algebra, matrices H are related to Gaussian quadrature on the unit circle, as well as direct and inverse unitary eigenvalue problems, and efficient algorithms for several problems can be found in [12,13,16,2,1], among others. In operator theory literature the structure in (1.5) is associated with the *Naimark dilation*, see, e.g., [7], Section 2.4 in [3], and Section 6.7 in [11]. Szegő polynomials can often be found in signal processing literature, because they describe the state-space structure for lattice digital filters, see, e.g., [30,24,23,34,21].

The paper is structured as follows. In the next section we recall the classical Björck–Pereyra algorithm for solving a classical Vandermonde linear system. In Section 3, after a brief bit of background the main result is presented, a recursive factorization of the inverse of a Szegő–Vandermonde matrix, and formulas for the resulting fast algorithm for solving the corresponding linear system. In Section 4 several interesting numerical experiments are performed, as the proposed algorithm yields good forward error results while solving ill-conditioned systems.

2. The classical Björck–Pereyra algorithm

Recall that in [6], the authors derive a representation for the inverse $V(x)^{-1}$ of an $n \times n$ Vandermonde matrix (1.1) as the product of bidiagonal matrices, that is,

$$V(x)^{-1} = U_1 \cdots U_{n-1} \cdot \tilde{L}_{n-1} \cdots \tilde{L}_1 \tag{2.1}$$

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

$$a = U_1 \cdots U_{n-1} \cdot \tilde{L}_{n-1} \cdots \tilde{L}_1 f \tag{2.2}$$

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 $O(n^3)$ operations in general. This favorable complexity results from the fact that the matrices U_k and \tilde{L}_k are sparse. More specifically, if $V(x)$ is given by (1.1), the factors U_k and \tilde{L}_k are given by

$$U_k = \left[\begin{array}{c|ccc} I_{k-1} & & & \\ \hline & 1 & -x_k & \\ & & 1 & \ddots \\ & & & \ddots & -x_k \\ & & & & 1 \end{array} \right], \tag{2.3}$$

$$\tilde{L}_k = \left[\begin{array}{c|ccc} I_k & & & \\ \hline & \frac{1}{x_{k+1}-x_1} & & \\ & & \ddots & \\ & & & \frac{1}{x_n-x_{n-k}} \end{array} \right] \cdot \left[\begin{array}{c|ccc} I_{k-1} & & & \\ \hline & 1 & & \\ & -1 & 1 & \\ & & \ddots & \ddots \\ & & & -1 & 1 \end{array} \right]. \tag{2.4}$$

In the next section we will present our algorithm for solving Szegő–Vandermonde systems by obtaining a counterpart of the above factorization.

3. Factorization formula

In order to derive an analog of the Björck–Pereyra algorithm for the Szegő case, we will use the concept of associated polynomials, defined next.

3.1. Associated (generalized Horner) polynomials

Following [22] define the *associated polynomials* $\hat{R} = \{\hat{r}_0(x), \dots, \hat{r}_n(x)\}$ for a given system of polynomials $R = \{r_0(x), \dots, r_n(x)\}$ satisfying $\text{degr}_k = k$ via the relation

$$\frac{r_n(x) - r_n(y)}{x - y} = [r_0(x) \quad r_1(x) \quad r_2(x) \quad \cdots \quad r_{n-1}(x)] \cdot \begin{bmatrix} \hat{r}_{n-1}(y) \\ \hat{r}_{n-2}(y) \\ \vdots \\ \hat{r}_1(y) \\ \hat{r}_0(y) \end{bmatrix}. \tag{3.1}$$

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

However, before proceeding we first clarify the existence of such polynomials. Firstly, the polynomials associated with the monomials exist. Indeed, if P is the system of $n + 1$ polynomials $P = \{1, x, x^2, \dots, x^{n-1}, r_n(x)\}$, then

$$\frac{r_n(x) - r_n(y)}{x - y} = \begin{bmatrix} 1 & x & x^2 & \dots & x^{n-1} \end{bmatrix} \cdot \begin{bmatrix} \hat{p}_{n-1}(y) \\ \hat{p}_{n-2}(y) \\ \vdots \\ \hat{p}_1(y) \\ \hat{p}_0(y) \end{bmatrix} = \sum_{i=0}^{n-1} x^i \cdot \hat{p}_{n-1-i}(y), \tag{3.2}$$

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

Secondly, 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 $\widehat{R} = \{\hat{r}_0(x), \hat{r}_1(x), \dots, \hat{r}_{n-1}(x), \hat{r}_n(x)\}$ (with $\hat{r}_n(x) = r_n(x)$) satisfying (3.1). 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

$$\begin{bmatrix} 1 & x & x^2 & \dots & x^{n-1} \end{bmatrix} S = \begin{bmatrix} r_0(x) & r_1(x) & r_2(x) & \dots & r_{n-1}(x) \end{bmatrix} \tag{3.3}$$

where S is an $n \times n$ upper triangular invertible matrix capturing the recurrence relations of the polynomial system R . Inserting SS^{-1} into (3.2) between the row and column vectors and using (3.3), we see that the polynomials associated with R are

$$\begin{bmatrix} \hat{r}_{n-1}(y) \\ \hat{r}_{n-2}(y) \\ \vdots \\ \hat{r}_1(y) \\ \hat{r}_0(y) \end{bmatrix} = S^{-1} \begin{bmatrix} \hat{p}_{n-1}(y) \\ \hat{p}_{n-2}(y) \\ \vdots \\ \hat{p}_1(y) \\ \hat{p}_0(y) \end{bmatrix}, \tag{3.4}$$

where $\widehat{P} = \{\hat{p}_0(x), \dots, \hat{p}_{n-1}(x)\}$ are the classical Horner polynomials and S is from (3.3).

3.2. Factorization formula

In this and subsequent sections we consider the Szegő polynomials $\Phi = \{\phi_k^\#(x)\}$, i.e. the polynomials orthogonal on the unit circle satisfying the two-term recurrence relations (1.4). The following lemma will be useful in finding the factorization formula below.

Lemma 1. *Let $\Phi = \{\phi_k^\#(x)\}_{k=0}^{n-1}$ be a system of Szegő polynomials corresponding to the reflection coefficients $\{\rho_k\}_{k=0}^n$ as defined in Section 1. For $k = 1, 2, \dots, n - 1$ denote by $\Phi^{(k)}$ the system of polynomials $\Phi^{(k)} = \{\hat{\phi}_0^{(k)}(x), \dots, \hat{\phi}_k^{(k)}(x)\}$ associated with the truncated system $\{\phi_0^\#(x), \dots, \phi_k^\#(x)\}$. Then*

$$\begin{aligned}
 & \begin{bmatrix} \hat{\phi}_0^{(1)}(x) & \hat{\phi}_1^{(2)}(x) & \cdots & \hat{\phi}_{n-2}^{(n-1)}(x) \\ & \hat{\phi}_0^{(2)}(x) & \cdots & \hat{\phi}_{n-3}^{(n-1)}(x) \\ & & \ddots & \vdots \\ & & & \hat{\phi}_0^{(n-1)}(x) \end{bmatrix}^{-1} \\
 &= \begin{bmatrix} \mu_1 & -x - \rho_2 \rho_1^* & -\rho_3 \mu_2 \rho_1^* & \cdots & -\rho_{n-1} \mu_{n-2} \cdots \mu_2 \rho_1^* \\ & \mu_2 & -x - \rho_3 \rho_2^* & \cdots & -\rho_{n-1} \mu_{n-2} \cdots \mu_3 \rho_2^* \\ & & \mu_3 & \ddots & \vdots \\ & & & \ddots & -x - \rho_{n-1} \rho_{n-2}^* \\ & & & & \mu_{n-1} \end{bmatrix}. \tag{3.5}
 \end{aligned}$$

Proof. It was shown in [25, Eq. (4.8)] that the n -term recurrence relations for the truncated associated polynomials $\hat{\Phi}^\#$ are

$$\begin{aligned}
 \mu_{k-m} \hat{\phi}_m^{(k)}(x) &= x \cdot \hat{\phi}_{m-1}^{(k)}(x) + \rho_{k-m+1} \rho_{k-m}^* \cdot \hat{\phi}_{m-1}^{(k)}(x) + \rho_{k-m+2} \mu_{k-m+1} \rho_{k-m}^* \cdot \hat{\phi}_{m-2}^{(k)} \\
 &+ \cdots + \rho_k \mu_{k-1} \cdots \mu_{k-m+1} \rho_{k-m}^* \cdot \hat{\phi}_0^{(k)}(x), \tag{3.6}
 \end{aligned}$$

for $m = 1, \dots, k - 1$, and also

$$\mu_k \hat{\phi}_0^{(k)} = 1. \tag{3.7}$$

Now consider the product

$$\begin{aligned}
 & \begin{bmatrix} \mu_1 - x - \rho_2 \rho_1^* & -\rho_3 \mu_2 \rho_1^* & \cdots & -\rho_{n-1} \mu_{n-2} \cdots \mu_2 \rho_1^* \\ \vdots & & & \vdots \\ \mu_i - x - \rho_{i+1} \rho_i^* & -\rho_{i+2} \mu_{i+1} \rho_i^* \cdots & -\rho_{n-1} \mu_{n-2} \cdots \mu_{i+1} \rho_i^* \\ \vdots & & & \vdots \\ \mu_{n-1} \end{bmatrix} \\
 & \times \begin{bmatrix} \hat{\phi}_0^{(1)}(x) & \cdots & \hat{\phi}_{j-1}^{(j)}(x) & \cdots & \hat{\phi}_{n-2}^{(n-1)}(x) \\ & & \hat{\phi}_{j-2}^{(j)}(x) & & \hat{\phi}_{n-3}^{(n-1)}(x) \\ & & \vdots & & \vdots \\ & & \hat{\phi}_0^{(j)}(x) & & \hat{\phi}_1^{(n-1)}(x) \\ & & & \ddots & \hat{\phi}_0^{(n-1)}(x) \end{bmatrix}, \tag{3.8}
 \end{aligned}$$

where the (i, j) entry of this product defined by the highlighted row and column can be seen as (3.6) with $k = j, m = j - i$ if $i \neq j$ and (3.7) with $k = i$ if $i = j$. Thus this product is the identity, implying (3.5). \square

Our algorithm involves a recursive factorization of the inverse matrix $V_R(x)^{-1}$. In the following we use the notation $x_{1:n} = (x_1, \dots, x_n)$, etc.

Lemma 2. Let $\Phi = \{\phi_k^\#(x)\}_{k=0}^{n-1}$ be a system of Szegő polynomials corresponding to the reflection coefficients $\{\rho_k\}_{k=0}^n$ and complementary parameters $\{\mu_k\}_{k=0}^n$ as defined in Section 1, and let $x_{1:n}$ be n distinct points. Then the inverse of $V_R(x_{1:n})$ admits a decomposition

$$V_R(x_{1:n})^{-1} = U_1 \cdot \begin{bmatrix} 1 & 0 \\ 0 & V_R(x_{2:n})^{-1} \end{bmatrix} L_1, \tag{3.9}$$

with

$$U_1 = \begin{bmatrix} \frac{1}{\mu_0} & -x_1 - \rho_1 \rho_0^* & -\rho_2 \mu_1 \rho_0^* & \cdots & \cdots & -\rho_{n-1} \mu_{n-2} \cdots \mu_1 \rho_0^* \\ & \frac{1}{\mu_1} & -x_1 - \rho_2 \rho_1^* & \cdots & \cdots & -\rho_{n-1} \mu_{n-2} \cdots \mu_2 \rho_1^* \\ & & \frac{1}{\mu_2} & \ddots & & -\rho_{n-1} \mu_{n-2} \cdots \mu_3 \rho_2^* \\ & & & \ddots & \ddots & \vdots \\ & & & & \frac{1}{\mu_{n-2}} & -x_1 - \rho_{n-1} \rho_{n-2}^* \\ & & & & & \frac{1}{\mu_{n-1}} \end{bmatrix}, \tag{3.10}$$

$$L_1 = \begin{bmatrix} 1 & & & & & \\ & \frac{1}{x_2 - x_1} & & & & \\ & & \ddots & & & \\ & & & \frac{1}{x_n - x_1} & & \\ & & & & & \end{bmatrix} \begin{bmatrix} 1 & & & & & \\ -1 & 1 & & & & \\ \vdots & & \ddots & & & \\ -1 & & & & 1 & \end{bmatrix}. \tag{3.11}$$

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

$$V_R(x_{1:n}) = \begin{bmatrix} 1 & & & & \\ 1 & 1 & & & \\ \vdots & & \ddots & & \\ 1 & & & 1 & \end{bmatrix} \cdot \begin{bmatrix} 1 & & & & \\ & x_2 - x_1 & & & \\ & & \ddots & & \\ & & & & x_n - x_1 \end{bmatrix} \cdot \left[\begin{array}{c|ccc} 1 & 0 & & \\ 0 & \bar{R} & & \end{array} \right] \cdot \left[\begin{array}{c|ccc} \phi_0^\#(x_1) & \phi_1^\#(x_1) & \cdots & \phi_{n-1}^\#(x_1) \\ \hline 0 & & & I \end{array} \right], \tag{3.12}$$

where the matrix $\bar{R} = \left[\frac{\phi_j^\#(x_{i+1}) - \phi_j^\#(x_1)}{x_{i+1} - x_1} \right]$ consists of divided differences. By the discussion above, associated with the system Φ is the system $\hat{\Phi} = \{\hat{\phi}_k(x)\}$. Following the notation of Lemma 1, denote by $\hat{\Phi}^{(k)} = \{\hat{\phi}_0^{(k)}(x), \dots, \hat{\phi}_k^{(k)}(x)\}$ the system of polynomials associated with the truncated system $\{\phi_0(x), \dots, \phi_k(x)\}$. By the definition of the associated polynomials we have for $k = 1, 2, \dots, n - 1$

$$\frac{\phi_k^\#(x) - \phi_k^\#(y)}{x - y} = [\phi_0^\#(x) \quad \phi_1^\#(x) \quad \phi_2^\#(x) \quad \cdots \quad \phi_{k-1}^\#(x)] \cdot \begin{bmatrix} \hat{\phi}_{k-1}^{(k)}(y) \\ \hat{\phi}_{n-2}^{(k)}(y) \\ \vdots \\ \hat{\phi}_1^{(k)}(y) \\ \hat{\phi}_0^{(k)}(y) \end{bmatrix} \\ = \sum_{i=0}^{k-1} \phi_i^\#(x) \cdot \hat{\phi}_{k-1-i}^{(k)}(y).$$

Finally, denoting by $\widehat{\Phi}^{(k)} = \{\widehat{\phi}_0^{(k)}(x), \dots, \widehat{\phi}_k^{(k)}(x)\}$ the system of polynomials associated with the truncated system $\{\phi_0(x), \dots, \phi_k(x)\}$ we can further factor \overline{R} as

$$\overline{R} = V_R(x_{2:n}) \cdot \begin{bmatrix} \widehat{\phi}_0^{(1)}(x_1) & \widehat{\phi}_1^{(2)}(x_1) & \cdots & \widehat{\phi}_{n-2}^{(n-1)}(x_1) \\ & \widehat{\phi}_0^{(2)}(x_1) & \cdots & \widehat{\phi}_{n-3}^{(n-1)}(x_1) \\ & & \ddots & \vdots \\ & & & \widehat{\phi}_0^{(n-1)}(x_1) \end{bmatrix}. \tag{3.13}$$

The last matrix on the right-hand side of (3.13) can be inverted by Lemma 1. Therefore, inverting (3.12) and substituting (3.5) results in (3.9). \square

3.3. New Björck–Pereyra type algorithm

Like the classical Björck–Pereyra algorithm, the recursive nature of the formula (3.9) allows a decomposition

$$V_R(x_{1:n})^{-1} = U_1 \cdots U_{n-1} \cdot L_{n-1} \cdots L_1, \tag{3.14}$$

with the upper and lower triangular factors given via recursively applying Lemma 2, arriving at

$$U_k = \left[\begin{array}{c|c} I_{k-1} & 0 \\ \hline 0 & \widetilde{U}_k \end{array} \right] = \left[\begin{array}{c|cccccccc} I_{k-1} & & & & & & & & \\ \hline & \frac{1}{\mu_0} - x_k - \rho_1 \rho_0^* & -\rho_2 \mu_1 \rho_0^* & \cdots & \cdots & -\rho_{n-k} \mu_{n-k-1} \cdots \mu_1 \rho_0^* & & & \\ & \frac{1}{\mu_1} & -x_k - \rho_2 \rho_1^* & \cdots & \cdots & -\rho_{n-k} \mu_{n-k-1} \cdots \mu_2 \rho_1^* & & & \\ & & \frac{1}{\mu_2} & \ddots & & -\rho_{n-k} \mu_{n-k-1} \cdots \mu_3 \rho_2^* & & & \\ & & & \ddots & \ddots & \vdots & & & \\ & & & & \frac{1}{\mu_{n-k-1}} & -x_k - \rho_{n-k} \rho_{n-k-1}^* & & & \\ & & & & & \frac{1}{\mu_{n-k}} & & & \end{array} \right], \tag{3.15}$$

$$L_k = \left[\begin{array}{c|cccc} I_{k-1} & & & & \\ \hline & 1 & & & \\ & & \frac{1}{x_{k+1} - x_k} & & \\ & & & \ddots & \\ & & & & \frac{1}{x_n - x_k} \end{array} \right] \left[\begin{array}{c|ccc} I_{k-1} & & & \\ \hline & 1 & & \\ & -1 & 1 & \\ & \vdots & & \ddots \\ & -1 & & 1 \end{array} \right]. \tag{3.16}$$

Remark 1. It is worth noting that there is a difference between the factors L_k in (3.16) and the factors \widetilde{L}_k in (2.4). From the uniqueness of the L factor in the LU -factorization, the formula is valid with either choice.

The associated linear system can be solved by multiplying (3.14) by the right-hand side vector f in the linear system $V_R(x_{1:n})x = f$. However, unlike the classical Björck–Pereyra algorithm,

the matrices U_k involved in the proposed algorithm are not sparse. The sparseness of these factors in the classical Vandermonde case (in fact they are even bidiagonal, see (2.3)) is exactly what reduces the complexity by an order of magnitude to $O(n^2)$. In fact, for a general polynomial system R , a similar derivation of an algorithm is possible, however the overall cost is again $O(n^3)$.

Although the matrices U_k are not sparse, a method of fast multiplication of U_k by a vector will allow the desired reduction in complexity. Denote by H_k the $k \times k$ leading submatrix of H given in (1.5) as in Section 1, and further denote

$$H_k(x) = H_k - xI; \tag{3.17}$$

that is, $H_k(x)$ is the leading $k \times k$ submatrix of H with entries on the main diagonal shifted by x . With this notation, by comparing (3.15) and (1.5), we make the observation that the matrix \tilde{U}_k given in (3.15) can be written as

$$U_k = \left[\begin{array}{c|c} I_{k-1} & 0 \\ \hline 0 & \tilde{U}_k \end{array} \right], \quad \tilde{U}_k = \left[\begin{array}{c|ccc} \frac{1}{\mu_0} & & & \\ 0 & & & \\ \vdots & & & \\ 0 & & & \\ \hline 0 & 0 & \dots & 0 & \frac{1}{\mu_{n-k}} \end{array} \right]. \tag{3.18}$$

This observation reduces the problem of fast multiplication of U_k by a vector to that of fast multiplication of H_k by a vector. Fast multiplication of H_k by a vector is easily achieved by using the well known decomposition of H_k as

$$H_k = G_k(\rho_1) \cdot G_k(\rho_2) \cdot \dots \cdot G_k(\rho_{k-1}) \cdot \tilde{G}_k(\rho_k), \tag{3.19}$$

where

$$G_k(\rho_j) = \text{diag} \left\{ I_{j-1}, \left[\begin{array}{cc} \rho_j & \mu_j \\ \mu_j & -\rho_j^* \end{array} \right], I_{k-j-1} \right\}, \quad j = 1, 2, \dots, k - 1 \tag{3.20}$$

and

$$\tilde{G}_k(\rho_k) = \text{diag}\{I_{k-1}, \rho_k\}, \tag{3.21}$$

see, for instance, [12,3], or [30].

Therefore, the factorization (3.19) reduces multiplication of H_k by a vector to $k - 1$ circular rotations, thus suggesting an efficient $O(n^2)$ implementation for our Björck–Pereyra like algorithm for Szegő–Vandermonde matrices.

4. Numerical Illustrations

To check the numerical performance of the proposed algorithm, the following numerical experiments were performed. All matrices used in these examples were 30×30 . The algorithm has been implemented in MATLAB version 7, which uses double precision. These results were compared with exact solutions calculated using the MATLAB Symbolic Toolbox command `vpa()`, which allows software-implemented precision of arbitrary numbers of digits. The number of digits was

set to 64, however in cases where the condition number of the coefficient matrix exceeded 10^{30} , this was raised to 100 digits to maintain accuracy.

In the tables, BP-SV denotes the proposed Björck–Pereyra like algorithm for Szegő–Vandermonde systems implemented using the results of the previous section. The factors L_k from (3.16) were used. GE indicates MATLAB’s Gaussian elimination. Finally, $\text{cond}(V)$ denotes the condition number of the matrix V computed via the MATLAB command $\text{cond}()$.

It is known (see [29,20]) 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*

$$|x_1| = \max_{1 \leq i \leq n} |x_i|, \quad \prod_{j=1}^{k-1} |x_k - x_j| = \max_{k \leq i \leq n} \prod_{j=1}^{k-1} |t_i - t_j|, \quad k = 2, \dots, n - 1 \tag{4.1}$$

(see [31,20,27,29]) 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 ordered randomly). A counterpart of this ordering is known for Cauchy matrices, see [5].

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

$$e = \frac{\|x - \hat{x}\|_2}{\|x\|_2} \tag{4.2}$$

where \hat{x} is the solution computed by each algorithm in MATLAB in double precision, and x is the exact solution.

Experiment 1. In Table 2, the values for $\rho_k, k = 1, \dots, n$ were chosen randomly (complex) in the unit disc, similarly for the entries of the right hand side vector $b_k, k = 1, \dots, n$. The nodes x_k were selected as the roots of the polynomial $\phi_n(x)$ defined by the reflection coefficients $\{\rho_k\}_{k=0}^n$. Choosing the parameters in this manner results in a (comparatively) well-conditioned matrix. As such, GE does well in this case, and the proposed algorithm does well provided the Leja ordering is used. Otherwise, its performance is not as good as GE. This demonstrates the usefulness of the Leja ordering. See Table 2.

Experiment 2. Next, the values for ρ_k were chosen randomly (complex) in the unit disc, similarly for the nodes x_k , and the entries of the right hand side vector b_k . Choosing such parameters, the condition number of the matrices is large, however, the BP-SV algorithm still produces excellent forward accuracy. Ten trial results are listed in Table 3.

Table 2
Random $\{\rho_k\}, \{b_k\}$ on the unit disc, $\{x_k\}$ roots of $\phi_n(x)$ corresponding to $\{\rho_k\}$

$\text{cond}(V)$	GEPP	BP-SV	BP-SV-L
9.7e+03	2.1e−15	5.1e−10	2.4e−15
8.1e+06	3.6e−14	3.9e−10	3.5e−15
2.4e+08	2.8e−13	3.4e−10	1.3e−15
8.5e+04	8.6e−15	2.3e−09	1.6e−15
3.3e+05	7.3e−14	9.7e−10	2.2e−15
3.1e+06	5.0e−14	3.3e−09	1.7e−15
1.6e+07	1.5e−14	4.0e−10	1.6e−15
3.6e+09	4.1e−13	7.7e−10	1.6e−15

Table 3

Random $\{\rho_k\}, \{x_k\}, \{b_k\}$ on the unit disc

cond(V)	GEPP	BP-SV	BP-SV-L
7.3e+14	4.0e-06	6.5e-15	3.7e-15
1.6e+15	1.8e-05	1.9e-15	7.2e-16
2.8e+15	4.6e-04	2.4e-15	7.5e-16
6.2e+16	6.8e-03	6.4e-16	8.0e-16
3.0e+17	5.9e-01	6.1e-15	3.3e-15
6.6e+17	6.7e-02	1.1e-15	1.3e-15
3.0e+18	1.4e+00	9.9e-16	9.2e-16
1.2e+19	9.9e-01	4.4e-13	4.4e-13

Table 4

Random $\{\rho_k\}, \{x_k\}, \{b_k\}$ on the unit disc, $\{\rho_k\}$ close to unit circle

Choice of ρ_k	cond(V)	GEPP	BP-SV	BP-SV-L
$.9 \leq \rho_k < 1$	3.4e+22	4.2e-02	1.1e-15	3.7e-15
	4.3e+22	5.1e-03	6.5e-15	9.5e-16
	1.1e+26	1.2e+00	1.8e-15	1.1e-15
	6.0e+26	2.1e-01	2.0e-15	5.2e-15
	6.1e+27	1.0e+00	4.3e-15	2.8e-15
$.99 \leq \rho_k < 1$	2.7e+36	5.7e-03	2.6e-15	1.8e-15
	8.6e+36	1.0e+00	4.0e-14	4.0e-14
	1.1e+37	2.6e+00	1.5e-15	7.7e-15
	1.6e+37	1.0e+00	2.0e-15	6.3e-16
	8.3e+37	6.0e-01	1.4e-15	8.5e-16
$.999 \leq \rho_k < 1$	3.3e+47	6.1e-05	2.3e-15	4.4e-16
	6.1e+48	4.1e+00	1.6e-15	4.4e-16
	6.8e+48	2.5e-01	2.3e-15	1.0e-15
	3.0e+51	9.9e-01	1.0e-15	1.2e-15
	9.6e+51	1.4e-01	3.8e-15	2.5e-15

Experiment 3. In the next experiment (Table 4), the values for x_k and b_k were chosen randomly in the unit disc, but the reflection coefficients are chosen randomly within the unit disc close to the unit circle. This has the effect of producing even more ill-conditioned matrices. The proposed algorithm still produces very good forward accuracy, as seen in Table 4.

The results of Experiments 1–4 are depicted in Fig. 1.

Experiment 4. In [8,4] it was shown that the behavior of BKO-type algorithms can depend on the direction of the right hand side vector. We include a similar experiment here where the outcome is consistent with observations in [8,4]. This is illustrated in Fig. 2, which shows the results given a fixed set of $\{\rho_k\}$ and $\{x_k\}$ on the unit circle, and the results of applying the various algorithms to solve the system with each (left) singular vector as the right hand side. It is observed that in this example there is a dependence of the performance of the BP-SV algorithm on the direction of the right hand side vector, as opposed to GE, as the accuracy of the algorithm significantly improves when passing from the first to the last singular vectors.

Conclusions. These initial numerical experiments indicate that the proposed Björck–Pereyra like algorithm can attain very good forward error compared to Gaussian elimination for even some ill-conditioned matrices. These observations are preliminary, and more practical experience may be needed. Additionally, an error analysis may provide more insights.

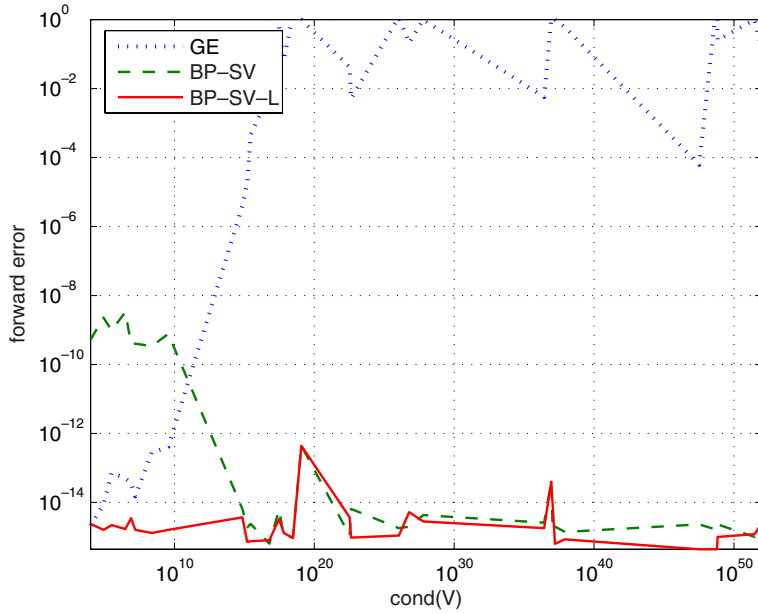


Fig. 1. Effects of conditioning.

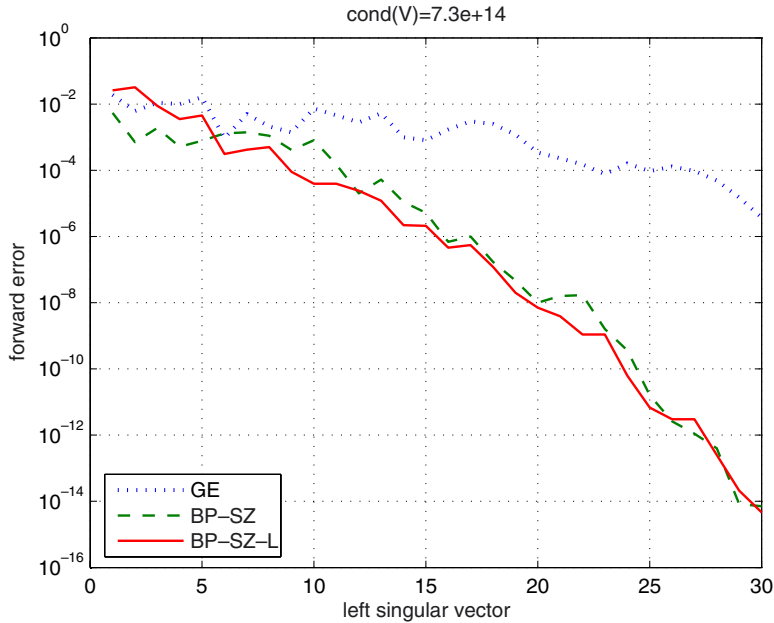


Fig. 2. Effects of different left singular vectors as right hand side.

5. Conclusions

In this paper an analog of the well known Björck–Pereyra algorithm was presented for Szegő–Vandermonde matrices. This algorithm was derived by exploiting the properties of the related

unitary Hessenberg matrix, which resulted in the small computational complexity $O(n^2)$ operations, as opposed to the usual $O(n^3)$ of standard (structure-ignoring) methods. Initial numerical experiments using this algorithm indicate good performance for ill-conditioned systems, in fact better results than Gaussian elimination for the same systems.

Acknowledgments

The authors would like to sincerely thank the anonymous referee for very carefully reading the paper, and for a number of suggestions that allowed us to improve the presentation.

References

- [1] G. Ammar, D. Calvetti, L. Reichel, Continuation methods for the computation of zeros of Szegő polynomials, *Linear Algebra Appl.* 249 (1996) 125–155.
- [2] G. Ammar, W. Gragg, L. Reichel, An analogue for the Szegő polynomials of the Clenshaw algorithm, *J. Comput. Appl. Math.* 46 (1993) 211–216.
- [3] M. Bakonyi, T. Constantinescu, *Schur's algorithm and several applications*, Pitman Research Notes in Mathematics Series, vol. 61, Longman Scientific and Technical, Harlow, 1992.
- [4] T. Boros, T. Kailath, V. Olshevsky, Fast Björck–Pereyra-type algorithm for parallel solution of Cauchy linear equations, *Linear Algebra Appl.* 302–303 (1999) 265–293.
- [5] T. Boros, T. Kailath, V. Olshevsky, Pivoting and backward stability of fast algorithms for solving Cauchy linear equations, Special issue on structured and infinite systems of linear equations, *Linear Algebra Appl.* 343/344 (2002) 63–99.
- [6] A. Björck, V. Pereyra, Solution of Vandermonde systems of equations, *Math. Comp.* 24 (1970) 893–903.
- [7] T. Constantinescu, On the structure of the Naimark dilation, *J. Operator Theory* 12 (1984) 159–175.
- [8] T. Chan, D. Foulser, Effectively well-conditioned linear systems, *SIAM J. Sci. Stat. Comput.* 9 (1988) 963–969.
- [9] D. Calvetti, L. Reichel, Fast inversion of Vandermonde-like matrices involving orthogonal polynomials, *BIT*, 1993.
- [10] G. Forney, *Concatenated Codes*, The MIT Press, Cambridge, 1966.
- [11] C. Foias, A.E. Frazho, *The Commutant Lifting Approach to Interpolation Problems*, Birkhauser-Verlag, 1989.
- [12] W.B. Gragg, Positive definite Toeplitz matrices, the Arnoldi process for isometric operators, and Gaussian quadrature on the unit circle (in Russian). In : E.S. Nikolaev (Ed.), *Numerical methods in Linear Algebra*, Moskow University Press, 1982, pp. 16–32. English translation in: *J. Comput. Appl. Math.* 46 (1993) 183–198.
- [13] W.B. Gragg, The QR algorithm for unitary Hessenberg matrices, *J. Comput. Appl. Math.* 16 (1986) 1–8.
- [14] L.Y. Geronimus, Polynomials orthogonal on a circle and their applications, *Am. Math. Translations* 3 (1954) 1–78. (Russian original 1948).
- [15] I. Gohberg, V. Olshevsky, Fast inversion of Chebyshev–Vandermonde matrices, *Numer. Math.* 67 (1) (1994) 71–92.
- [16] W.B. Gragg, L. Reichel, A divide and conquer method for unitary and orthogonal eigenproblems, *Numer. Math.* 57 (1990) 695–718.
- [17] U. Grenader, G. Szegő, *Toeplitz Forms and Applications*, University of California Press, 1958.
- [18] G. Golub, C. van Loan, *Matrix Computations*, third ed., Johns Hopkins University Press, 1996.
- [19] N.J. Higham, Error analysis of the Björck–Pereyra algorithms for solving Vandermonde systems, *Numer. Math.* 50 (1987) 613–632.
- [20] N.J. Higham, Stability analysis of algorithms for solving confluent Vandermonde-like systems, *SIAM J. Matrix Anal. Appl.* 11 (1) (1990) 23–41.
- [21] H. Kimura, Generalized Schwartz form and lattice-ladder realizations for digital filters, *IEEE Trans. Circ. Syst.* 32 (11) (1985) 1130–1139.
- [22] T. Kailath, V. Olshevsky, Displacement structure approach to polynomial Vandermonde and related matrices, *Linear Algebra Appl.* 261 (1997) 49–90.
- [23] T. Kailath, B. Porat, State-space generators for orthogonal polynomials, in: V. Mandrekar, H. Salehi (Eds.), *Prediction Theory and Harmonic Analysis*, The Pesu Masani Volume, North-Holland Publishing Company, 1983, pp. 131–163.
- [24] M. Morf, D.T. Lee, State-space structure of ladder canonical forms, in: *Proc. 18th Conf. on Control and Design*, December 1980, pp. 1221–1224.

- [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) 37–67.
- [26] V. Olshevsky, Associated polynomials, unitary Hessenberg matrices and fast generalized Parker–Traub and Björck–Pereyra algorithms for Szegő–Vandermonde matrices, in: D. Bini, E. Tyrtyshnikov, P. Yalamov (Eds.), *Structured Matrices: Recent Developments in Theory and Computation*, NOVA Science Publ., 2001, pp. 67–78.
- [27] V. Olshevsky, Pivoting for structured matrices and rational tangential interpolation, in: *Fast Algorithms for Structured Matrices: Theory and Applications*, CONM/323, AMS Publications, 2003, pp. 1–75.
- [28] F. Parker, Inverses of Vandermonde matrices, *Amer. Math. Monthly* 71 (1964) 410–411.
- [29] L. Reichel, Newton interpolation at Leja points, *BIT* 30 (1990) 23–41.
- [30] P.A. Regalia, *Adaptive IIR Filtering in Signal Processing and Control*, Marcel Dekker, New York, 1995.
- [31] L. Reichel, G. Opfer, Chebyshev–Vandermonde systems, *Math. Comp.* 57 (1991) 703–721.
- [32] J. Traub, Associated polynomials and uniform methods for the solution of linear problems, *SIAM Rev.* 8 (3) (1966) 277–301.
- [33] E.E. Tyrtyshnikov, How bad are Hankel matrices?, *Numer. Math.* 67 (1994) 261–269.
- [34] M. Takizawa, Hisao Kishi, N. Hamada, Synthesis of lattice digital filter by the state space variable method, *Trans. IECE Jpn.* J65-A (1983) 363–370.