Numerical Algorithms **33:** 153–161, 2003.

On the evaluation of polynomial coefficients

Daniela Calvetti [a,*] and Lothar Reichel [b,**]

[a] *Department of Mathematics, Case Western Reserve University, Cleveland, OH 44106, USA*
E-mail: dxc57@po.cwru.edu
[b] *Department of Mathematical Sciences, Kent State University, Kent, OH 44242, USA*
E-mail: reichel@math.kent.edu

Received 24 May 2002; accepted 2 September 2002

Dedicated to Claude Brezinski on the occasion of his 60th birthday.

The evaluation of the coefficients of a polynomial from its zeros is considered. We show that when the evaluation is carried out by the standard algorithm in finite precision arithmetic, the accuracy of the computed coefficients depends on the order in which the zeros are introduced. An ordering that enhances the accuracy for many polynomials is presented.

Keywords: polynomial coefficients, power basis, round-off error, Leja ordering

AMS subject classification: 65Q05, 65G50

1. Introduction

Polynomials play a central role in computational and applied mathematics; see [3] for a historical review. The determination of the zeros $\{z_j\}_{j=0}^{n-1}$ of a monic polynomial of degree n,

$$p_n(z) = z^n + \sum_{j=0}^{n-1} a_{j,n} z^j, \tag{1.1}$$

is a classical problem of computational mathematics, which continues to receive considerable attention; see, e.g., [1,2,4, and references therein], as well as the bibliography [7]. This note is concerned with the inverse problem: Given the zeros $\{z_j\}_{j=0}^{n-1}$ of a monic polynomial p_n of degree n, determine the coefficients $\{a_{j,n}\}_{j=0}^{n-1}$ in the representation (1.1). The standard approach to solving the latter problem is to recursively compute, for increasing values of k, the coefficients $\{a_{j,k}\}_{j=0}^{k-1}$ in the power-form representation of

* Research supported in part by NSF grant DMS-0107841.
** Research supported in part by NSF grant DMS-0107858.

the polynomials

$$p_k(z) = \prod_{j=0}^{k-1}(z - z_j) = z^k + \sum_{j=0}^{k-1} a_{j,k} z^j, \quad k = 1, 2, \ldots, n.$$

This approach yields algorithm 1.1 below. We denote the coefficients computed by algorithm 1.1 using finite-precision arithmetic by $\tilde{a}_{j,k}$. Hence, $\tilde{a}_{j,k}$ is the computed approximation of $a_{j,k}$, and we would like the error $\tilde{a}_{j,n} - a_{j,n}$ to be small.

Algorithm 1.1 (Computation of coefficients).
Input: Zeros $\{z_j\}_{j=0}^{n-1}$;
Output: Computed coefficients $\{\tilde{a}_{j,n}\}_{j=0}^{n-1}$;
$\tilde{a}_{0,1} := -z_0$;
for $k := 1, 2, \ldots, n-1$ **do**
  $\tilde{a}_{k,k+1} := \tilde{a}_{k-1,k} - z_k$;
  for $j := 1, 2, \ldots, k-1$ **do**
    $\tilde{a}_{j,k+1} := \tilde{a}_{j-1,k} - z_k \tilde{a}_{j,k}$;
  end j;
  $\tilde{a}_{0,k+1} := -z_k \tilde{a}_{0,k}$;
end k;

The coefficients $\tilde{a}_{j,k}$ for fixed j and $k = j+1, j+2, \ldots, n$ can be stored in the same storage location. Algorithm 1.1 therefore can be implemented so that only $O(n)$ storage locations are required. This is the scheme implemented by the Matlab function poly.

The following example shows that the coefficients computed by algorithm 1.1 may suffer from low accuracy. All computations of the present paper were carried out in Matlab on a personal computer with machine epsilon $2 \cdot 10^{-16}$. We used the Matlab implementation poly of algorithm 1.1. In the computed examples, we display the quantities

$$\varepsilon_n = \max_{0 \leqslant j < n} |\tilde{a}_{j,n} - a_{j,n}|, \tag{1.2}$$

$$\rho_n = \max_{0 \leqslant j < k \leqslant n} |\tilde{a}_{j,k}|. \tag{1.3}$$

Example 1.1. The zeros of the polynomial

$$p_n(z) = z^n - 1 \tag{1.4}$$

are given by

$$z_j = \exp\left(\frac{2\pi \mathrm{i} j}{n}\right), \quad j = 0, 1, \ldots, n-1, \tag{1.5}$$

where $\mathrm{i} = \sqrt{-1}$. We refer to the order of the zeros (1.5) as monotonic because the argument of the zeros z_j increases monotonically with j.

Table 1
Example 1.1. Roots of unity ordered monotonically.

n	ρ_n	ε_n
32	$4.1 \cdot 10^{3}$	$6.1 \cdot 10^{-10}$
64	$9.1 \cdot 10^{7}$	$6.4 \cdot 10^{-2}$
128	$6.1 \cdot 10^{16}$	$1.0 \cdot 10^{15}$

Table 2
The bit-reversal function $b_3(j)$.

j	0	1	2	3	4	5	6	7
$b_3(j)$	0	4	2	6	1	5	3	7

Table 1 displays the error (1.2) of the coefficients computed by algorithm 1.1. The poor accuracy in the computed coefficients $\tilde{a}_{j,n}$ depends on that some coefficients $\tilde{a}_{j,k}$ are of large magnitude, but the coefficients $a_{j,n}$ of the polynomials (1.4) are not. The largest magnitude of the coefficients $\tilde{a}_{j,k}$ is measured by ρ_n, defined by (1.3), which is displayed in table 1.

The loss of accuracy encountered in example 1.1 can be reduced by reordering the zeros (1.5) before using them in algorithm 1.1. Section 2 discusses an ordering scheme that is applicable when the zeros $\{z_j\}_{j=0}^{n-1}$ are roots of unity with n a power of 2. The ordering scheme of section 2 suggests a method for ordering arbitrary zeros. The latter ordering method is described in section 3.

2. Ordering of roots of unity

This section considers the polynomials of example 1.1 in some more detail. Let $n = 2^{\ell}$ for some positive integer ℓ, and let the integer $j \in [0, n-1]$ have the binary representation

$$j = \sum_{r=0}^{\ell-1} \beta_r 2^r, \quad \beta_r \in \{0, 1\}.$$

Introduce the bit-reversal function

$$b_\ell(j) = \sum_{r=0}^{\ell-1} \beta_r 2^{\ell-1-r},$$

which reverses the order of the ℓ bits in the binary representation of j. Values of $b_3(j)$ are displayed in table 2.

Introduce the zeros $\hat{z}_j$, obtained by reordering the zeros (1.5) using the bit-reversal function $b_\ell(j)$, i.e.,

$$\hat{z}_j = z_{b_\ell(j)}, \quad 0 \leqslant j < n, \quad n = 2^{\ell}. \tag{2.1}$$

Note that while the value of the bit-reversal function $b_\ell(j)$ for fixed $j < 2^\ell$ depends on the value of ℓ, the location of the point $\hat{z}_j$ in the complex plane does not. In fact, the Matlab code used for example 2.1 below computes the arguments of the points $\hat{z}_j$ instead of the values of the bit-reversal function. The arguments for the points (2.1) can be determined in $O(n)$ arithmetic floating point operations.

Lemma 2.1. Let ℓ be a positive integer, and let the integer $k \in [1, 2^\ell - 1]$ have the binary representation

$$k = \sum_{r=0}^{\ell-1} \beta_r 2^r, \quad \beta_r \in \{0, 1\}. \tag{2.2}$$

Let the zeros $\hat{z}_j$ be defined by (2.1). Then there are coefficients γ_r of unit magnitude, such that

$$\prod_{j=0}^{k-1}(z - \hat{z}_j) = \prod_{\{r:\beta_r=1\}} \left(z^{2^r} - \gamma_r\right), \tag{2.3}$$

where the product in the right-hand side is over all values r, such that the coefficient β_r equals to one in the binary representation (2.2) of k.

Proof. The result follows from [5, lemma 2.3], where also explicit expressions for the coefficients γ_r can be found. The factorization in the right-hand side of (2.3) is independent of the value of ℓ, provided that $k < 2^\ell$. □

Theorem 2.2. Let $n = 2^\ell$ for some positive integer ℓ. Assume that k is an integer, such that $0 < k \leqslant n$, and let the points $\hat{z}_j$, $0 \leqslant j < n$, be defined by (2.1). Introduce the polynomials

$$\hat{p}_k(z) = \prod_{j=0}^{k-1}(z - \hat{z}_j), \quad k = 1, 2, \ldots, n.$$

The coefficients in the representation of the polynomials $\hat{p}_k$ in power form

$$\hat{p}_k(z) = z^k + \sum_{j=0}^{k-1} \hat{a}_{j,k} z^j, \quad k = 1, 2, \ldots, n,$$

satisfy

$$\max_{0 \leqslant j < k \leqslant n} |\hat{a}_{j,k}| = 1. \tag{2.4}$$

Proof. Multiplying the factors in the product in the right-hand side of (2.3) gives a representation of $\hat{p}_k$ in power form. Due to the special form of the factors, it is easy to see that no coefficient in the power form representation is of magnitude larger than one and that the nonvanishing coefficients are of magnitude one. □

Table 3
Example 2.1. Roots of unity ordered by bit-reversal.

n	ρ_n	ε_n
32	1	$4.8 \cdot 10^{-15}$
64	1	$9.5 \cdot 10^{-15}$
128	1	$1.7 \cdot 10^{-14}$

Table 4
Example 2.1. Roots of unity ordered randomly. The minimum, average and maximum values of ρ_n and ε_n are over 1000 experiments.

n	ρ_n			ε_n		
	min	average	max	min	average	max
32	$3.2 \cdot 10^{0}$	$3.9 \cdot 10^{1}$	$8.3 \cdot 10^{2}$	$4.2 \cdot 10^{-15}$	$4.6 \cdot 10^{-13}$	$5.4 \cdot 10^{-11}$
64	$1.2 \cdot 10^{0}$	$8.8 \cdot 10^{2}$	$4.6 \cdot 10^{4}$	$5.8 \cdot 10^{-14}$	$4.5 \cdot 10^{-10}$	$7.2 \cdot 10^{-8}$
128	$6.1 \cdot 10^{1}$	$1.6 \cdot 10^{5}$	$2.7 \cdot 10^{7}$	$2.9 \cdot 10^{-12}$	$2.0 \cdot 10^{-5}$	$1.2 \cdot 10^{-2}$

Example 2.1. This example considers the same sets of zeros as example 1.1, but introduces the zeros in algorithm 1.1 in different orders. This results in higher accuracy of the computed coefficients. Table 3 displays values of ρ_n and ε_n when the zeros (1.5) are ordered by bit-reversal, i.e., when the zeros are ordered according to (2.1). The table shows that $\rho_n = 1$, in agreement with theorem 2.2. The small value of ρ_n helps make the absolute errors in the computed coefficients $\{\tilde{a}_{j,n}\}_{j=0}^{n-1}$ small.

We also applied algorithm 1.1 to randomly ordered roots of unity, i.e., the zeros (1.5) are chosen in random order. Table 4 displays the minimum, average and maximal values of ρ_n and ε_n obtained over 1000 experiments. The table shows the average values of ρ_n and ε_n to grow much faster with n than the corresponding quantities when the zeros are ordered by bit-reversal. Comparing tables 3 and 4 with table 1 shows that zeros ordered by bit-reversal as well as randomly ordered zeros give much smaller errors ε_n than monotonically ordered zeros.

3. Ordering of arbitrary zeros

This section discusses ordering of arbitrarily distributed zeros in the complex plane. Algorithm 3.1 below determines the so-called Leja ordering of the zeros. The algorithm assumes that the zeros are distinct; however, this restriction easily can be removed, see below.

Algorithm 3.1 (Leja ordering of zeros).
Input: Set $\mathbb{K} = \{z_j\}_{j=0}^{n-1}$ of distinct zeros;
Output: Set $\{\hat{z}_j\}_{j=0}^{n-1}$ of Leja ordered zeros;
Let $\hat{z}_0$ be a zero in the set $\mathbb{K}$, such that $\hat{z}_0 = \max_{z\in\mathbb{K}} |z|$;

for $k := 1, 2, \ldots, n-1$ **do**
Determine $\hat{z}_k \in \mathbb{K}$, so that

$$\prod_{j=0}^{k-1} |\hat{z}_k - \hat{z}_j| = \max_{z \in \mathbb{K}} \prod_{j=0}^{k-1} |z - \hat{z}_j|;$$

end k;

We refer to the order determined by algorithm 3.1 as the Leja order, because when the set $\mathbb{K}$ is a compact continuum in the complex plane, the points determined by the algorithm are known as Leja points; see [6] for a definition of Leja points and for some of their properties. Algorithm 3.1 can be implemented to require only $O(n^2)$ arithmetic floating point operations. When the zeros in the set $\mathbb{K}$ in algorithm 3.1 are not all distinct, zeros of higher multiplicity are ordered consecutively. The following theorem shows that the Leja order is related to the bit-reversal order.

Theorem 3.1. The set $\{\hat{z}_j\}_{j=0}^{k-1}$ of points defined by (2.1) is Leja ordered.

Proof. The theorem can be shown by using lemma 2.1. □

The relation between Leja and bit-reversal order suggests that using Leja ordered zeros in algorithm 1.1 may be beneficial for the accuracy in the computed coefficients. This is illustrated by the following examples.

Example 3.1. Let the zeros be $n = 2^\ell$ roots of unity for $\ell = 5, 6, 7$ as in example 1.1. Leja ordering of these sets of zeros yields $\rho_{2^\ell} = 1$ for $\ell = 5, 6, 7$ and $\varepsilon_{32} = 4.4 \cdot 10^{-15}$, $\varepsilon_{64} = 8.7 \cdot 10^{-15}$, $\varepsilon_{128} = 1.7 \cdot 10^{-15}$. Thus, the errors in the computed coefficients are as small as when the roots of unity are ordered by bit-reversal; cf. table 3.

Example 3.2. This example illustrates the performance of Leja, monotonic and random ordering of n roots of unity when n is not a power of 2. Table 5 shows that the errors in the computed coefficients are small when the zeros are Leja ordered. Note that, differently from example 3.1, $\rho_n \neq 1$ for Leja ordered roots of unity. Table 5 also illustrates

Table 5
Example 3.2. Roots of unity in Leja and monotonic orders.

n	Leja order		Monotonic order	
	ρ_n	ε_n	ρ_n	ε_n
31	1.1	$4.4 \cdot 10^{-15}$	$3.1 \cdot 10^{3}$	$3.7 \cdot 10^{-10}$
63	1.2	$9.3 \cdot 10^{-15}$	$6.6 \cdot 10^{7}$	$2.8 \cdot 10^{-2}$
127	1.5	$1.6 \cdot 10^{-14}$	$4.4 \cdot 10^{16}$	$1.1 \cdot 10^{15}$

Table 6
Example 3.2. Roots of unity ordered randomly. The minimum, average and maximum values of ρ_n and ε_n are over 1000 experiments.

n	ρ_n			ε_n		
	min	average	max	min	average	max
31	$2.7 \cdot 10^{0}$	$3.5 \cdot 10^{1}$	$8.1 \cdot 10^{2}$	$4.5 \cdot 10^{-15}$	$3.1 \cdot 10^{-13}$	$1.9 \cdot 10^{-11}$
63	$1.2 \cdot 10^{1}$	$1.0 \cdot 10^{3}$	$2.6 \cdot 10^{5}$	$5.1 \cdot 10^{-14}$	$9.2 \cdot 10^{-10}$	$3.7 \cdot 10^{-7}$
127	$1.1 \cdot 10^{2}$	$2.8 \cdot 10^{5}$	$4.0 \cdot 10^{7}$	$3.7 \cdot 10^{-12}$	$2.3 \cdot 10^{-5}$	$7.8 \cdot 10^{-3}$

Table 7
Example 3.3. Zeros of Chebyshev polynomials in Leja and monotonic orders.

n	Leja order		Monotonic order	
	ρ_n	ε_n	ρ_n	ε_n
32	$9.3 \cdot 10^{1}$	$1.2 \cdot 10^{-13}$	$4.1 \cdot 10^{2}$	$7.1 \cdot 10^{-12}$
64	$2.8 \cdot 10^{4}$	$5.4 \cdot 10^{-11}$	$5.6 \cdot 10^{5}$	$1.3 \cdot 10^{-4}$
128	$3.4 \cdot 10^{9}$	$1.4 \cdot 10^{-5}$	$1.5 \cdot 10^{12}$	$2.9 \cdot 10^{8}$

the rapid growth of the error in the computed coefficients when the roots of unity are ordered monotonically.

Table 6 displays the growth of the computed coefficients when the roots of unity are ordered randomly. The table shows the smallest, average and largest values of ρ_n and ε_n over 1000 experiments. Note that the average values of ε_n displayed in table 6 grow much faster with n than the values of ε_n obtained with Leja ordered roots of unity.

Example 3.3. The sets $\{z_j\}_{j=0}^{n-1}$ in this example are the zeros of monic Chebyshev polynomials $T_n(z) = 2^{1-n}\cos(n \arccos(z))$, i.e.,

$$z_j = \cos\left(\frac{2j+1}{2n}\pi\right), \quad j = 0, 1, \ldots, n-1. \tag{3.1}$$

The coefficients $a_{j,n}$ of T_n expressed in terms of the power basis are explicitly known, see, e.g., [8, p. 4], and are used for determining the errors in the coefficients $\tilde{a}_{j,n}$ computed by algorithm 1.1.

Table 7 displays the errors in the computed coefficients $\tilde{a}_{j,n}$ when the zeros z_j are ordered monotonically, i.e, when they are ordered according to (3.1), and when they are Leja ordered. The latter ordering is seen to give higher accuracy than monotonic ordering.

Table 8 shows the errors in the computed coefficients when the zeros are ordered randomly. The average errors obtained are about the same as the errors achieved by Leja ordering. Also, the values of ρ_n obtained by random ordering of the zeros are about the same as the values achieved by Leja ordering. We therefore do not display the former.

Table 8
Example 3.3. Randomly ordered zeros of Chebyshev polynomials. The minimum, average and maximum values of ε_n are over 1000 experiments.

n	ε_n		
	min	average	max
32	$7.1 \cdot 10^{-14}$	$1.3 \cdot 10^{-13}$	$2.4 \cdot 10^{-12}$
64	$2.5 \cdot 10^{-11}$	$5.8 \cdot 10^{-11}$	$1.7 \cdot 10^{-10}$
128	$8.6 \cdot 10^{-6}$	$1.4 \cdot 10^{-5}$	$2.0 \cdot 10^{-5}$

Table 9
Example 3.4. Zeros distributed equidistantly on 2 concentric circles in Leja and monotonic orders.

n	Leja order		Monotonic order	
	ρ_n	ε_n	ρ_n	ε_n
75	1.1	$6.6 \cdot 10^{-15}$	$1.1 \cdot 10^{6}$	$1.6 \cdot 10^{-5}$
90	1.2	$7.0 \cdot 10^{-15}$	$2.6 \cdot 10^{7}$	$6.4 \cdot 10^{-3}$

Table 10
Example 3.4. Randomly ordered zeros distributed equidistantly on 2 concentric circles. The minimum, average and maximum values of ρ_n and ε_n are over 1000 experiments.

n	ρ_n			ε_n		
	min	average	max	min	average	max
75	$6.4 \cdot 10^{0}$	$5.4 \cdot 10^{2}$	$7.0 \cdot 10^{4}$	$2.4 \cdot 10^{-14}$	$3.4 \cdot 10^{-10}$	$1.5 \cdot 10^{-7}$
90	$1.1 \cdot 10^{1}$	$1.5 \cdot 10^{3}$	$1.9 \cdot 10^{5}$	$4.3 \cdot 10^{-14}$	$1.6 \cdot 10^{-9}$	$6.2 \cdot 10^{-7}$

Thus, in this example Leja ordering and random ordering give on the average about the same absolute errors in the computed coefficients. These errors are larger than the corresponding errors in the computed coefficients of example 3.1. This depends on the fact that for high degrees, the Chebyshev polynomials have coefficients of large magnitude. The tabulated values of ρ_n for the Leja ordering of the zeros are accurate approximations of $\max_{1 \leqslant j < n} |a_{j,n}|$.

Example 3.4. Let $\frac{2}{3}n$ of the zeros be distributed equidistantly on the unit circle and let the remaining $\frac{1}{3}n$ zeros be allocated equidistantly on a circle of radius $\frac{1}{2}$ centered at the origin, i.e.,

$$z_j = \begin{cases} \exp\left(\dfrac{3\pi i j}{n}\right), & j = 0, 1, \ldots, \dfrac{2}{3}n - 1, \\ \dfrac{1}{2}\exp\left(\dfrac{6\pi i j}{n}\right), & j = \dfrac{2}{3}n, \dfrac{2}{3}n + 1, \ldots, n - 1. \end{cases} \tag{3.2}$$

The coefficients $a_{j,n}$ of the polynomial p_n either vanish or are of unit magnitude. We refer to the ordering (3.2) as monotonic. Table 9 shows the accuracy of the computed

coefficients when the zeros are Leja and monotonically ordered. The former ordering is seen to give significantly higher accuracy.

Table 10 displays ρ_n and ε_n when the zeros are ordered randomly. A comparison with table 9 shows that Leja ordering gives the highest accuracy and monotonic ordering the lowest.

The examples of the present paper, as well as numerous other computed examples, indicate that bit-reversal ordering and Leja ordering of the zeros can give significantly higher accuracy of the computed coefficients of polynomials in power form than monotonic ordering. Since Leja ordering can be applied to arbitrary sets of zeros, we propose that it be used in conjunction with algorithm 1.1.

References

[1] G.S. Ammar, D. Calvetti, W.B. Gragg and L. Reichel, Polynomial zerofinders based on Szegő polynomials, J. Comput. Appl. Math. 127 (2001) 1–16.

[2] A. Bini and G. Fiorentino, Design, analysis and implementation of a multiprecision polynomial rootfinder, Numer. Algorithms 23 (2000) 127–173.

[3] C. Brezinski, Historical perspective on interpolation, approximation and quadrature, in: *Handbook of Numerical Analysis*, Vol. 3, eds. P.G. Ciarlet and J.L. Lions (North-Holland, Amsterdam, 1994) pp. 3–46.

[4] D. Calvetti, L. Reichel and F. Sgallari, A modified companion matrix method based on Newton polynomials, in: *Fast Algorithms*, ed. V. Olshevsky, Contemporary Mathematics (Amer. Math. Soc., Providence, RI) to appear.

[5] B. Fischer and L. Reichel, A stable Richardson iteration method for complex linear systems, Numer. Math. 54 (1988) 225–242.

[6] F. Leja, Sur certaines suits liées aux ensemble plan et leur application à la representation conforme, Ann. Polon. Math. 4 (1957) 8–13.

[7] J.M. McNamee, An updated supplementary bibliography on roots of polynomials, J. Comput. Appl. Math. 110 (1999) 305–306. The actual bibliography is available at the Web site `http://www.elsevier.nl/locate/cam`.

[8] T.J. Rivlin, *Chebyshev Polynomials*, 2nd ed. (Wiley, New York, 1990).