# A New Approach to Fast Polynomial Interpolation and Multipoint Evaluation

Victor Pan*, Akimou Sadikou†, Elliott Landowne†,
and Olen Tiga‡

TR-92-055

August 1992

## Abstract

The fastest known algorithms for the problems of polynomial evaluation and multipoint interpolation are devastatingly unstable numerically because of their recursive use of polynomial divisions. We apply a completely distinct approach to compute approximate solutions to both problems equally fast but with improved numerical stability. Our approach relies on new techniques, so far not used in this area: we reduce the problems to Vandermonde matrix computations and then exploit some recent methods for improving computations with structured matrices.

Keywords: polynomial evaluation, polynomial interpolation, dense structured matrices, numerical stability, fast approximate algorithms.

1991 Mathematics Subject Classification: 65D05, 65D15, 65F30, 68Q25, 15A06.

The authors were supported by NSF Grant CCR-9020690 and PSC CUNY Awards 661340, 662478, 668541 and 669210.

* Mathematics and Computer Science Department, Lehman College, CUNY, Bronx, NY 10468

† Ph.D. Program in Computer Science, CUNY, Graduate School, 33 West 42nd St., New York, NY 10036

‡ Ph.D. Program in Mathematics, CUNY, Graduate School, 33 West 42nd St., New York, NY 10036

# A New Approach to Fast Polynomial Interpolation and Multipoint Evaluation

Victor Pan [*]
Mathematics and Computer Science Department
Lehman College CUNY, Bronx, NY 10468

Akimou Sadikou [*] and Elliott Landowne [*]
Ph.D. Program in Computer Science, CUNY [†]

Olen Tiga [*]
Ph.D. Program in Mathematics, CUNY [†]

**Abstract**

The fastest known algorithms for the problems of polynomial evaluation and multipoint interpolation are devastatingly unstable numerically because of their recursive use of polynomial divisions. We apply a completely distinct approach to compute approximate solutions to both problems equally fast but with improved numerical stability. Our approach relies on new techniques, so far not used in this area: we reduce the problems to Vandermonde matrix computations and then exploit some recent methods for improving computations with structured matrices.

*Keywords:* polynomial evaluation, polynomial interpolation, dense structured matrices, numerical stability, fast approximation algorithms.

*1991 Mathematics Subject Classification:* 65D05, 65D15, 65F30, 68Q25, 15A06.

## 1 Introduction

It is known that interpolation and evaluation (on any set of $n$ nodes) of an $n$-th degree polynomial can be performed in $O(n \log^2 n)$ (arithmetic) operations [1], but recursive application of several polynomial divisions is involved, and this makes the resulting algorithms highly unstable numerically (thus, restricting their application to the case of computers performing exact rational arithmetic). Fast Fourier transform (FFT) enables us to solve both of these problems in $O(n \log n)$ operations [1] with no numerical stability problems [10], in the special case where the nodes are roots of unity, and similarly in some other special cases [4]. In particular, for the input nodes lying on a bounded real interval, the estimate $O(n \log^2 n)$ can be improved [14,17]. However, for a general set of nodes on the complex plane, the order of $n^2$ operations is required in the known numerically stable solutions.

In this paper we propose a new approach to both problems, which we solve approximately, within a given tolerance $K\epsilon$ to the error, $K$ denoting the condition number of some auxiliary computational problem [see (4.1) in section 4]. Solution time is proportional to $n(\log^2 n + \log(1/\epsilon))$, which turns into $O(n \log^2 n)$ if $\log(1/\epsilon) = O(\log^2 n)$. Polynomial divisions are avoided in this approach, based on the application of computations with structured matrices.

The solution involves Toeplitz type linear systems, which for many inputs may still cause some numerical stability problems [7], not as devastating, however, as ones caused by recursive polynomial divisions. In particular, we may shift to symmetrized systems which are still of Toeplitz type, and if they remain sufficiently well-conditioned after their symmetrization, then numerical stability problems are avoided [7].

[*]Supported by NSF Grant CCR-9020690 and PSC CUNY Awards 661340, 662478, 668541 and 669210.

[†]Address: Graduate School, 33 West 42nd Street, New York, NY 10036

Our algorithms have the feature of many iterative algorithms: their output errors and their running time decrease with the condition number of the auxiliary linear system (in our case of Toeplitz type) to which we reduce the solution. Furthermore, we may try to use a random transformation of some input parameters in order to improve the condition of the latter linear system (see remark 3.1 and the derivation of the complexity estimates based on our first algorithm of section 4).

It is known that the Vandermonde matrix (defining the interpolation problem) is ill-conditioned for a very large class of sets of interpolation nodes, and the parameter $K$ of (4.1), defining the approximation error of our computations, tends to be large. Thus we cannot, as of now, recommend our algorithms for practical computations, except for the special cases of node sets defining well-conditioned Vandermonde matrices.

From the theoretical point of view, however, the new algorithms may be of interest since they demonstrate some previously hidden correlations between computations with polynomials and with structured matrices. Specifically, we represent the original computational problems of polynomial evaluation and interpolation in the form of operations with a Vandermonde matrix. Then we apply the techniques of [13] for computations with structured matrices and reduce the original problem with any set of nodes to the case of roots of unity as the nodes; in this case FFT applies. The reduction involves Toeplitz type computations (with matrices having displacement rank at most 3) and a single multiplication of a generalized Hilbert matrix by a vector, at which stage we apply the fast approximation algorithm of [16], known to be effective and reliable in numerous computations, in particular, for integral equations and n-body mechanics.

We present and analyze our algorithms in section 4 after some preliminaries in sections 2 (definitions) and 3 (auxiliary results).

## 2 Definitions

Let $n$ be a positive integer, and $i$,$j$,$k$ integer parameters, ranging from 0 to $n-1$. Matrix rows and vector components are represented by $i$, columns by $j$. $W^T$ is the transpose of a matrix (vector) $W$; $W^H$ is the Hermitian transpose of $W$.

Complex vectors $\mathbf{e}$, $\mathbf{f}$, $\mathbf{g}$, $\mathbf{r}$, $\mathbf{u}$, $\mathbf{v}$, $\mathbf{w}$, $\mathbf{x}$ and $\mathbf{y}$ are of the form $\mathbf{h} = [h_0, \ldots, h_{n-1}]^T$ (where $\mathbf{h}$ can represent any listed vector). $r_i = r^i$, where $r = \exp(2\pi\sqrt{-1}/n)$, is a primitive n-th root of 1. Let the components $u_i$ of $\mathbf{u}$ be pairwise distinct and not equal to integer powers of r.

Matrices $I$, $J$, $V$, $H$, $T$ and $Z$ are of size $n \times n$. In particular, the matrices

$$J = \begin{bmatrix} 0 & & & 1 \\ & & \cdot & \\ & \cdot & & \\ 1 & & & 0 \end{bmatrix} \text{ (reversion) and } Z = \begin{bmatrix} 0 & & & 0 \\ 1 & \cdot & & \\ & \cdot & \cdot & \\ & & \cdot & \cdot \\ 0 & & 1 & 0 \end{bmatrix} \text{ (displacement)}$$

satisfy $J^2 = I$ (the identity matrix), $J\mathbf{u} = [u_{n-1}, \ldots, u_0]^T$, $Z\mathbf{u} = [0, u_0, \ldots, u_{n-2}]^T$.

$V(\mathbf{u}) = [u_i^j]$ is a Vandermonde matrix. $V(\mathbf{r}) = [r^{ij}]$ is the matrix of discrete Fourier transform (DFT) on n points, so that $V(\mathbf{r})\mathbf{u} = \left[\sum_{i=0}^{n-1} r^{ij} u_i\right]^T$ is the DFT of a vector $\mathbf{u}$.

$H(\mathbf{u}, \mathbf{v}) = \left[\dfrac{1}{u_i - v_j}\right]$ is a generalized Hilbert matrix ($u_i \neq v_j$ for all $i$ and $j$) .

$T = [t_{n-1+i-j}]$ is a Toeplitz matrix.

$$TJ = [t_{i+j}], \quad JT = [t_{2n-2-i-j}] \tag{2.1}$$

are Hankel matrices.

[Toeplitz (Hankel) matrices have entries that are invariant under shifts in the diagonal (antidiagonal) direction].

## 3 Auxiliary Results

We recall the following simple and/or well-known results (which we state by using definitions of section 2):

$$H(\mathbf{u},\mathbf{v}) = -H^T(\mathbf{v},\mathbf{u}). \tag{3.1}$$

$$V(\mathbf{r}) = V^T(\mathbf{r}), \quad V^H(\mathbf{r})V(\mathbf{r}) = nI. \tag{3.2}$$

**Fact 3.1** *The values of a polynomial $w(x) = \sum_{i=0}^{n-1} w_i x^i$ (with coefficient vector $\mathbf{w}$) on the set of points $u_0, \ldots, u_{n-1}$ are given by the vector*

$$\mathbf{v} = V(\mathbf{u})\mathbf{w}. \tag{3.3}$$

Fact 3.1 defines the problem of interpolation and multipoint evaluation of a polynomial $w(x)$ in terms of a vector equation (compare Problems 4.1 and 4.2).

**Remark 3.1** It is simple to shift from polynomial $w(x)$ to $t(x) = w(ax + b)$ for any fixed complex numbers $a$ and $b$, and vice versa, at the cost of $O(n \log n)$ operations [2]. Even simpler is the transition to the reverse polynomial $w_{\text{rev}}(x) = x^n w(1/x)$. These transformations enable us to vary (to our convenience) the input matrix $V(\mathbf{u})$ of the problems of polynomial interpolation and multipoint evaluation. For evaluation, we may also vary the input node set, for instance, by partitioning it into two or several subsets, complementing each subset to $n$ points at our choice and solving two or several evaluation problems.

**Fact 3.2** [1]. *Given vectors $\mathbf{u}$ and $\mathbf{r}$, it suffices to use $O(n \log n)$ arithmetic operations to compute $V(\mathbf{r})\mathbf{u}$ and $V^H(\mathbf{r})\mathbf{u}$ (that is, to perform the forward and inverse DFT of a vector $\mathbf{u}$).*

**Fact 3.3** *a)* [1] *Given a vector $\mathbf{v}$ and a Toeplitz matrix $T$, it suffices to use $O(n \log n)$ arithmetic operations to compute the vector $T\mathbf{v}$.*

*b)* [3,5,6,11,12] *Furthermore, $O(n \log^2 n)$ arithmetic operations suffice to compute $T^{-1}\mathbf{v}$ if $T$ is nonsingular.*

**Fact 3.4** *The estimates of the previous fact hold even if the matrix $T$ is replaced by any matrix of the form $V^T(\mathbf{u})V(\mathbf{u})$, $W(\mathbf{u},\mathbf{r}) = V^T(\mathbf{r})H(\mathbf{r},\mathbf{u})V(\mathbf{u})$ or $W^T(\mathbf{u},\mathbf{r})$.*

**Proof.** The extension of fact 3.3 to the matrix $V^T(\mathbf{u})V(\mathbf{u}) = \left[\sum_{k=0}^{n-1} u_k^{i+j}\right]$ follows since this is a Hankel matrix [compare (2.1)]. The extension of fact 3.3 to $W(\mathbf{u},\mathbf{r})$ and $W^T(\mathbf{u},\mathbf{r})$ follows from [13]. (Specifically, proposition 6.1 of [13] implies that $W(\mathbf{u},\mathbf{r})$, $W^T(\mathbf{u},\mathbf{r})$ are Toeplitz type matrices defined with their $n \times 3$ displacement generator matrices (see definitions in [9,13]), and to such matrices both parts of fact 3.3 can be extended [see [9] and/or [12] on the extension of part a) and [5] or [12] on the extension of part b)]).□

**Propositon 3.5** [16]. *Given a natural $n$, positive $a$, $q$, $s$, and $\epsilon$ and three complex vectors $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{y}$ of dimension $n$, it suffices to use $(5n-1)s$ arithmetic operations to compute, within the error bound $\epsilon$, the values $(H(\mathbf{v},\mathbf{u})\mathbf{y})_i = \sum_{k=0}^{n-1} y_k/(v_i - u_k)$, for $i = 0, 1, \ldots, n-1$, provided that*

$$s \geq \frac{\log(an) - \log((1-q)\epsilon)}{\log(1/q)}, \tag{3.4}$$

$$a \geq |y_k/u_k|, \quad 1 > q > |v_i/u_k| \quad \textit{for all } i \textit{ and } k. \tag{3.5}$$

**Proof** (see appendix).□

**Remark 3.2** *a) If, say, $q < 1/2$, $\log a = O(\log n)$, $\log(1/\epsilon) = O(\log n)$, then (3.4) can be satisfied for $s = O(\log n)$.*

*b) In our algorithms of the next section, $\mathbf{v} = \mathbf{r}$, and the vector $\mathbf{u}$ can be linearly transformed, according to remark 3.1. This enables us to insure the last inequality of (3.5) for $q = 1/2$ and for all $i$ and $k$.*

## 4 Interpolation and Multipoint Evaluation

**Problem 4.1** *Interpolation.*

**Input:** *vectors* $\mathbf{u}$ *and* $\mathbf{v}$.

**Output:** *vector* $\mathbf{w}$ *satisfying* (3.3).

Note that, by assumption about vectors $\mathbf{u}$ and $\mathbf{r}$ (in section 2), $V(\mathbf{u})$ is non-singular, and $H(\mathbf{r},\mathbf{u})$ can be defined. Let $H(\mathbf{r},\mathbf{u})$ be non-singular (until the end of this section).

**Solution:** successively compute the three vectors:

1. $\mathbf{f} = H(\mathbf{r},\mathbf{u})\mathbf{v}$,
2. $\mathbf{g} = V^T(\mathbf{r})\mathbf{f} = V(\mathbf{r})\mathbf{f}$ [see (3.2)],
3. $\mathbf{w} = W^{-1}(\mathbf{u},\mathbf{r})\mathbf{g}$.

The *correctness* of this algorithm follows since for the computed vectors $\mathbf{w}$ and $\mathbf{g}$, we have:

$$W(\mathbf{u},\mathbf{r})\mathbf{w} = W(\mathbf{u},\mathbf{r})W^{-1}(\mathbf{u},\mathbf{r})\mathbf{g} = \mathbf{g} = V^T(\mathbf{r})H(\mathbf{r},\mathbf{u})\mathbf{v},$$

and (3.3) follows since $V(\mathbf{r})$ and $H(\mathbf{r},\mathbf{u})$ are nonsingular.

We apply the algorithm for approximate evaluation of $\mathbf{w}$, so as to decrease the estimated *computational complexity*:

At stage 1, approximating the components of $\mathbf{f}$ (within $\epsilon > 0$) requires $(5n-1)s$ operations, for $s$ defined by (3.4);

Stage 2 requires $O(n \log n)$ operations (see fact 3.2);

Finally, at stage 3, we need to compute the matrix $W(\mathbf{u},\mathbf{r})$, of Toeplitz type, or more precisely, to compute its displacement generator of length (at most) 3. For this, we just need to compute the products $\mathbf{w}(\ell) = \left(W(\mathbf{u},\mathbf{r}) - ZW(\mathbf{u},\mathbf{r})Z^T\right)\mathbf{v}(\ell)$, $\ell = 1,2,3$, for three general vectors $\mathbf{v}(1)$, $\mathbf{v}(2)$, $\mathbf{v}(3)$.

Moreover, we may choose these vectors in the form $\mathbf{v}(\ell) = \mathbf{b}(\ell) = [1, b(\ell), (b(\ell))^2, \ldots, (b(\ell))^{n-1}]^T$, where $b(\ell)$ is a random parameter, $\ell = 1,2,3$. [Indeed, the $n \times n$ matrix $B = [(b(i))^j]$ has rank $n$, if all the $b(i)$ are distinct numbers, $i,j = 0,1,\ldots,n-1$; therefore, with a high probability (see [18,19]), for a random choice of $b(i)$, the matrix $\left(W(\mathbf{u},\mathbf{r}) - ZW(\mathbf{u},\mathbf{r})Z^T\right)B$ has rank 3, and so has any of its random $n \times 3$ submatrices]. The vector $V(\mathbf{u})\mathbf{b}(\ell)$, $\ell = 1,2,3$, can be computed in $O(n \log n)$ operations, since the evaluation of $V(\mathbf{u})\mathbf{b}(\ell)$ amounts to evaluation of the polynomial $\dfrac{1-(b(\ell)x)^n}{1-b(\ell)x} = \sum_{i=0}^{n-1}(b(\ell)x)^i$ at the points $u_0, \ldots, u_{n-1}$.

Therefore, the complexity of approximate evaluation of the vectors $\mathbf{w}(\ell)$, for $\ell = 1,2,3$, is still within the bounds of $O(n(s + \log n))$, for $s$ defined by (3.4), provided that $\epsilon$ denotes the tolerance to the errors of the appproximate multiplications of $H(\mathbf{r},\mathbf{u})$ by vectors in the process of the evaluation of $\mathbf{w}(\ell)$.

Given $\mathbf{w}(\ell)$, $\ell = 1,2,3$, we can, by using the algorithms of facts 3.3 or 3.4, find $W^{-1}\mathbf{g}$ at the cost $O(n \log^2 n)$. Therefore, stage 3 can be done at the cost of $O(n(s + log^2 n))$, $s$ defined by (3.4).

The bound $\epsilon$ on the approximation errors of all the multiplications of vectors by the matrix $H(\mathbf{u},\mathbf{r})$ is not substantially magnified in the subsequent multiplications of the resulting vectors by the matrix $V(\mathbf{r})$ (having 2-norm equal to 1) but may be substantially increased in the evaluation of $W^{-1}\mathbf{g}$ unless

$$K = \operatorname{cond} W \tag{4.1}$$

is small.

**Problem 4.2** *Evaluation.*
**Input:** *vectors* $\mathbf{u}$ *and* $\mathbf{w}$.
**Output:** *vector* $\mathbf{v}$ *satisfying* (3.3).

**Solution:** successively compute

1. $U(\mathbf{u}) = V^T(\mathbf{u})V(\mathbf{u})$,
2. $\mathbf{e} = U(\mathbf{u})\mathbf{w}$,
3. $\mathbf{x} = W^T(\mathbf{u},\mathbf{r})^{-1}\mathbf{e}$,
4. $\mathbf{y} = V(\mathbf{r})\mathbf{x}$,
5. $\mathbf{v} = H^T(\mathbf{r},\mathbf{u})\mathbf{y}$.

*Correctness.*
Premultiply both sides of (3.3) by $V^T(\mathbf{u})$ and obtain that

$$\mathbf{e} = V^T(\mathbf{u})V(\mathbf{u})\mathbf{w} = V^T(\mathbf{u})\mathbf{v}.$$

Then substitute $\mathbf{v} = H^T(\mathbf{r},\mathbf{u})\mathbf{y} = H^T(\mathbf{r},\mathbf{y})V(\mathbf{r})\mathbf{x}$, obtain that

$$W^T(\mathbf{u},\mathbf{r})\mathbf{x} = \mathbf{e},$$

and thus verify the correctness of the above solution.
*Complexity.*
Follow [8] to perform stage 1 in $O(n\log^2 n)$ operations. Specifically, first compute at this cost [1] the coefficients of the polynomial $\prod_{k=0}^{n}(u-u_k)$. Then obtain the power sums $\sum_{k=0}^{n-1} u_k^s$ in $O(n\log n)$ operations from the system of Newton's identities (see e.g. [15], appendix A).

We need $O(n\log n)$ operations at stage 2 and $O(n(s+\log^2 n))$ at stage 3, for $s$ defined in Proposition 3.5 [use fact 3.4 and the algorithm for the evaluation of $W(\mathbf{u},\mathbf{r})$ shown above], and $O(n\log n)$ operations at stage 4 (due to fact 3.2).

At stage 5, we approximate all the components of the vector $\mathbf{v}$ within the error bound $\epsilon$ at the cost $(5n-1)s$, where $s$ is defined by Proposition 3.5.

## Appendix: proof of Proposition 3.5

Substitute the expressions

$$\frac{1}{(v-u_k)} = -\frac{1}{u_k}\sum_{h=0}^{\infty}\left(\frac{v}{u_k}\right)^h$$

and obtain the power series representation

$$p(v) = \sum_{h=0}^{\infty} p_h v^h = \sum_{k=0}^{n-1} y_k/(v-u_k) = -\sum_{k=0}^{n-1}\frac{y_k}{u_k}\frac{1}{1-(v/u_k)}, \qquad (A.1)$$

$$p_h = -\sum_{k=0}^{n-1} y_k/u_k^{h+1}, \quad h = 0, 1, \ldots \quad . \qquad (A.2)$$

$p(v)$ converges when $|v|$ is small enough. Approximate $p(v)$ by the $s$-term partial sum and estimate the error in terms of $s$, $|v_i/u_k|$ and $a$. Due to (3.4) and (3.5), we obtain:

$$\left|p(v_i) - \sum_{h=0}^{s-1} p_h v_i^h\right| \le anq^s/(1-q) \le \epsilon, \quad \text{for all } i \qquad (A.3)$$

It remains to compute first $p_0, \ldots, p_{s-1}$ of (A.2), by using $(3n-1)s$ operations, and then $\sum_{h=0}^{s-1} p_h v_i^h$ for $i = 0, 1, \ldots, s-1$, by using $2ns$ operations. □

## References

[1] Aho, A., J. Hopcroft, and J. Ullman, *The Design and Analysis of Computer Algorithms* (Addison-Wesley, Reading, MA, 1974).

[2] Aho, A., K. Steiglitz, and J. Ullman, Evaluating Polynomials at Fixed Sets of Points, *SIAM J. on Computing* 4 (1975), 533-539.

[3] Ammar, G. S., and W. G. Gragg, Superfast Solution of Real Positive Definite Toeplitz Systems, *SIAM J. on Matrix Analysis and Applications* 9 (1988), 61-76.

[4] Bini, D., and V. Pan, *Numerical and Algebraic Computations with Matrices and Polynomials* (Birkhauser, Boston, to appear).

[5] Bitmead, R. R., and B.D.O. Anderson, Asymptotically Fast Solution of Toeplitz and Related Systems of Linear Equations, *Linear Algebra and Its Applications* 34 (1980), 103-116.

[6] Brent, R. P., F. G. Gustavson, and D. Y. Y. Yun, Fast Solution of Toeplitz Systems of Equations and Computations of Padé Approximations, *J. of Algorithms* 1 (1980), 259-295.

[7] Bunch, J. R., Stability of Methods for Solving Toeplitz Systems of Equations, *SIAM J. on Scientific and Statist. Computing* 6 (1985), 349-364.

[8] Canny, J.F., E. Kaltofen, and Y. Lakshman, Solving Systems of Non-linear Polynomial Equations Faster, *Proc. ACM-SIGSAM Int. Symp. on Symb. and Algebraic Computing* (1989), 121-128.

[9] Chun, J., T. Kailath, and H. Lev-Ari, Fast Parallel Algorithm for QR-Factorization of Structured Matrices, *SIAM J. of Scient. and Stat. Computing* 8 (1987), 899-913.

[10] Gentleman, W., and G. Sande, Fast Fourier Transforms for Fun and Profit, *Proc. AFIPS Fall Joint Comput. Conf.* 29 (1966), 563-578.

[11] de Hoog, H., On the Solution of Toeplitz Systems, *Linear Algebra and Its Applications* 88/89 (1987), 899-913.

[12] Musicus, B. R., Levinson and Fast Choleski Algorithms for Toeplitz and Almost Toeplitz Matrices, Internal Report, M.I.T. Res. Lab. for Electronics, 1981.

[13] Pan, V., Computations with Dense Structured Matrices, *Math. of Computations* 55 (1990), 179-190.

[14] Pan, V., Approximate Evaluation of a Polynomial on a Set of Real Points, Tech. Report, International Computer Science Institute, Berkeley, CA (1992).

[15] Pan, V., Parallel Least-Square Solution of General and Toeplitz-like Linear Systems, *Proc. 2nd Ann. ACM Symp. on Parallel Algorithms and Architecture* (1990), 244-253.

[16] Rokhlin, V., Rapid Solution of Integral Equations of Classical Potential Theory, *J. of Comput. Physics* 60 (1985), 187-207.

[17] Rokhlin, V., A Fast Algorithm for the Discrete Laplace Transformation, *J. of Complexity* 4 (1988), 12-32.

[18] Schwartz, J. T., Fast Probabilistic Algorithms for Verification of Polynomial Identities, *J. of ACM* 27 (1980), 701-717.

[19] Zippel, R. E., Probabilistic Algorithms for Sparse Polynomials, . Springer Lecture Notes in Computer Science 72 (1979), 216-226.