Approximate Polynomial Decomposition*

Robert M. Corless†‡ Mark W. Giesbrecht‡ David J. Jeffrey† Stephen M. Watt‡†

‡Dept. of Computer Science and †Dept. of Applied Mathematics
University of Western Ontario
London, ON, N6A 5B7, Canada

email: {Rob.Corless, Mark.Giesbrecht, David.Jeffrey, Stephen.Watt}@uwo.ca

1 INTRODUCTION

We consider approximately known polynomials $f(z) \in \mathbb{C}[z]$ or $f(z) \in \mathbb{R}[z]$ and examine the problem of functional decomposition. That is, given f, we wish to compute polynomials g and h such that

$$(f + \Delta f)(z) = (g \circ h)(z) = g(h(z)),$$

where $\deg g < \deg f$, $\deg h < \deg f$, $\deg \Delta f \leq \deg f$ and Δf is "small" with respect to the 2-norm of the vector of coefficients. In practice if $\|f\|$ denotes the 2-norm of f, then we compute g and h such that $\|\Delta f\|$ is a local minimum with respect to variations in g and h.

This problem has been studied for exact polynomials and rational functions by several authors [1, 2, 8, 10, 15, 16]. There are several reasons why approximate polynomial decomposition interests us:

- Decomposition is a fundamental operation on polynomials. Posing a natural, well-defined interpretation of approximate polynomial decomposition and presenting an algorithm for its computation further advances the program to develop a full collection of symbolic-numeric algorithms for polynomials.
- Sometimes one knows *a priori* from the problem domain that polynomials should be compositions. This can occur when modelling a phenomenon which comprises a number of sequential algebraic steps, for example, the positions of a multiply articulated robot arm.
- The decomposed form of a polynomial can be substantially less expensive to evaluate than either an expanded or factored form. For example, a dense polynomial of degree n would take approximately $2n$ operations to evaluate in either expanded or factored form. A presentation as two composition factors, however, would take between $4\sqrt{n}$ and n arithmetic operations.

The main results of this paper are:

(a) an iterative method to compute a decomposition of a given approximate polynomial, given a starting point. The iteration scheme, which is linearly convergent, is analogous to quotient-divisor iteration for the approximate GCD problem [5]. Further, each iteration can be executed with $O(n \log^2 n)$ floating point operations.

(b) a theorem showing that a surprisingly good starting point is obtained from the initial step of the exact algorithm, except when the leading coefficient of f is too small.

(c) an experimental comparison of the method with Newton iteration.

2 DEFINITIONS AND DESIGN CHOICES

Notation. We write f to indicate the polynomial operator $f = z \to f_0 + f_1 z + \cdots + f_n z^n$, and its value at a as $f(a)$. We write the transpose of the vector of coefficients of f as $\boldsymbol{f}^t = [f_0, f_1, \ldots, f_n]$. By $\boldsymbol{f}^*$ we mean the conjugate transpose of $\boldsymbol{f}$. By signum(α) for $0 \neq \alpha \in \mathbb{C}$ we mean $\alpha/|\alpha|$. Henceforth let $\deg f = n$, $\deg g = m$ and $\deg h = d$. We write $[\,z^\ell\,](p)$ for the coefficient of z^ℓ in p, following [7].

In this paper, the size of polynomials, and hence the distance between two polynomials, will be measured using the 2-norm. This norm can also be usefully expressed as a contour integral through Parseval's theorem (see [9]).

Lemma 2.1. *For a polynomial $f(z) = \sum_{k=0}^{n} f_k z^k \in \mathbb{C}[z]$,*

$$\|f\|^2 = \|f(z)\|^2 = |f_0|^2 + |f_1|^2 + \cdots + |f_n|^2, \quad (1)$$
$$= \frac{1}{2\pi}\int_0^{2\pi} f(e^{it})\overline{f}(e^{-it})\,dt \quad (2)$$
$$= \frac{1}{2\pi i}\int_C f(z)\overline{f}(1/z)\frac{dz}{z} \quad (3)$$
$$= [\,z^0\,]\ f(z)\overline{f}(1/z)\,, \quad (4)$$

where $z = e^{it}$ parameterizes C, the unit circle.

This norm has the following advantages.

(a) It allows us to evaluate partial derivatives of the norm in terms of polynomial and series manipulations. These can be used to express a sequence of least squares problems, whose solutions usually converge to a minimum perturbation $||\Delta f|| = ||f - g \circ h||$. The derivatives can also be used for Newton's method.

*Supported in part by the Natural Sciences and Engineering Research Council of Canada.

Appears in Proceedings of ISSAC'99: ACM International Symposium on Symbolic and Algebraic Computation, 1999, pp. 213-219

(b) Minimizing $\|\Delta f\|$ gives a near-Chebyshev minimum on the unit disk [13].

(c) It permits fast algorithms for the solution of subproblems at each iteration.

The expression of $\|f\|$ in the form (2) emphasizes the importance of the size of the values of $f(z)$ on the unit disk. This highlights the need for the following assumptions regarding the formulation of the problem:

(a) The location of the origin has been chosen (thus making explicit an implied assumption in previous numerical polynomial algorithms),

(b) The scale of $|z|$ has been chosen.

In particular, we assume that the problem context precludes a change of variable by an affine transformation $z \to bz + a$.

Remark. There is also a purely computational reason for avoiding such transformations, as is set out in the next theorem.

Theorem 2.2. *Shifting from z to $z - a$ can amplify any uncertainties in the coefficients of f by an amount as much as $(1 + |a|)^n/\sqrt{n+1}$ in norm. This is exponential in n, for any $a \neq 0$. Moreover, the relative uncertainties in each coefficient can be amplified by arbitrarily large amounts.*

In other words, such shifts are ill-conditioned.

Proof. By examining the condition of the matrix that determines the Taylor coefficients of the shifted polynomial

$$f_a(z) = \sum_{k=0}^{n} \frac{f^{(k)}(a)}{k!}(z-a)^k ,$$

one quickly finds the worst case for perturbation in the 1-norm: choose $f(z) = z^n$. Then $\|f\| = \|f\|_1 = 1$, but

$$f_a(z) = \sum_{k=0}^{n} \binom{n}{k} (-a)^k z^k$$

and hence $\|f_a\|_1 = (1 + |a|)^n$. Since the 1- and 2-norms are equivalent, with $\|f_a\| \geq \|f_a\|_1/\sqrt{n+1}$, the result follows.

In addition, changing a to $a + \Delta a$ will change $[\,z^k\,](f_a)$ by $\Delta a \cdot f^{(k+1)}(a)/f^{(k)}(a)$, which can be an arbitrarily large relative change. □

For completeness, we observe that any inner multiplicative scaling, $f_b = f \circ (z \to b \cdot z)$, has a purely diagonal matrix and relative uncertainties in each coefficient are not changed; but, as stated, we presume this scale has already been chosen.

However, a simple outer multiplicative scaling, given by $f \mapsto f/(\mathrm{signum}(f_n) \cdot \|f\|)$, is in fact desirable for determining an initial estimate to a decomposition (we assume that $f_n + \Delta f_n$ and f_n have the same sign—a perturbation that pushed the leading coefficient through zero would not likely be desirable). In the context of an application, this scaling is merely a convenience for the user: we measure changes in f relative to the original size of f. Since

$$\begin{aligned} f + \Delta f &= g \circ h = \sum_{\ell=0}^{m} g_\ell h^\ell, \\ \frac{f + \Delta f}{\mathrm{signum}(f_n)\|f\|} &= \sum_{\ell=0}^{m} \left(\frac{g_\ell}{\mathrm{signum}(f_n)\|f\|} \right) h^\ell \\ &= \sum_{\ell=0}^{m} \left(\frac{g_\ell}{\beta^\ell \cdot \mathrm{signum}(f_n)\|f\|} \right) (\beta h)^\ell , \end{aligned}$$

we may choose β so that $g_m/(\beta^m \mathrm{signum}(f_n)\|f\|) = 1$, making g monic. Alternatively, we could choose β to make $\|h\| = 1$, or even to make $\|g\| = 1$ by choosing β as the unique positive root of a certain polynomial; but we can only choose one of the three alternatives. Since the size of changes in f, *i.e.* $\|\Delta f\|$, should be measured with respect to $\|f\|$, it makes sense to insist that $\|f\| = 1$ (by scaling f); the only reason to insist on $\|g\| = 1$ or $\|h\| = 1$ would be to avoid overflow. This does not outweigh the convenience of $g_m = 1$ for Theorem 3.1 or the consequence $0 < f_n = g_m h_d^m = h_d^m$, which makes the size of the leading coefficient of h very simply related to the size of the leading coefficient of f. As an aside, we *could* insist that $\|f + \Delta f\| = 1$, but this would lead to pointless complications.

The computed decomposition $g \circ h$ yields a *particular* $f + \Delta f$. We observe that the set of all compositions of degree m and d polynomials is an $m + d + 2$-dimensional submanifold $\mathcal{S}$ of the set of degree $n = md$ polynomials. Our method identifies a point on $\mathcal{S}$ whose distance to f is locally a minimum. One might alternatively ask for the *closest* point in $\mathcal{S}$ to f, or for some point of $\mathcal{S}$ in a neighbourhood of f having some additional properties (for example, Δf having the same support as f). The manifolds have lower dimension when one accounts for the normalization $\|f\| = 1$ and g being monic.

3 INITIAL APPROXIMATION TO h

In the exact arithmetic case, the algorithm of Kozen & Landau [10] (see also [8]) is based on the observation that the leading d coefficients of f depend only on h and the degree of g, not on g itself ($f = h^m + g_{m-1}h^{m-1} + \cdots + g_0$). This was reformulated as a power series computation by von zur Gathen [15] by expansion at infinity: if $\widetilde{f}(z) = z^{\deg f} f(1/z)$ is the reciprocal polynomial of f, then $\widetilde{h}(z) = \widetilde{f}^{1/m}(z) \bmod z^{d+1}$ is the reciprocal polynomial of a composition factor of f of degree d (if such exists). That is, we compute the mth root of the power series expansion of f at $1/z$ and truncate the terms of positive degree, to obtain the expansion of h at $1/z$. Once h is found, identification of g requires only the solution of a linear system. If this system is inconsistent, there is no decomposition.

We use this same idea to give us an initial approximation $h^{(0)}$ for h in the case of an approximate f. One would expect this to work well provided that the trailing coefficients of g are "not too large".

The only substantive difference from the exact arithmetic setting is that f is not monic, so

$$\begin{aligned} \widetilde{f}(z) &= f_n + f_{n-1}z + f_{n-2}z^2 + \cdots f_0 z^m \\ &\Longrightarrow \widetilde{h}^{(0)}(z) = f_n^{1/m} + h_{d-1}z + \cdots + h_0 z^d \end{aligned}$$

and for definiteness we choose the positive (real) root for h_d (recall that we scaled f so that $\|f\| = 1$ with leading coefficient greater than 0).

On the quality of the initial approximation

Let h_{min} and g_{min} be the functions that minimize $\|\Delta f\|$. Further, let $f + \Delta f_{min} = g_{min} \circ h_{min}$. A bound showing the quality of $h^{(0)}$ is given by the next theorem.

Theorem 3.1. *There exists a constant K, depending on m and on the leading coefficient of f, such that if $\|\Delta f_{min}\|$ is small enough, then $\|h_{min} - h^{(0)}\| \le K \cdot \|\Delta f_{min}\|$.*

Proof. The Kozen-Landau and von zur Gathen algorithms define $h^{(0)}$ as the solution of the following system of nonlinear equations:

$$\begin{aligned} h_d^m - f_n &= 0 \quad h_d > 0 \\ h_{d-1} - F_{d-1}(h_d, f_{n-1}) &= 0 \\ h_{d-2} - F_{d-2}(h_{d-1}, h_d, f_{n-2}) &= 0 \\ &\vdots \\ h_0 - F_0(h_1, h_2, \dots, h_d, f_{n-d}) &= 0 \end{aligned}$$

for some functions F_k, derived from $[\, z^k \,] (f(z) - h^m(z)) = 0$ for $n - d \le k \le n$. Indeed these functions are known; for example, $F_{d-1}(h_d, f_{n-1}) = f_{n-1}/(m h_d^{m-1})$. Since $f_n > 0$, the roots of this nonlinear system are simple. That is, the Jacobian matrix, which is lower triangular, is non-singular at the root (in fact, the determinant is just $m^{d+1} f_n^d$).

Therefore the Jacobian and its inverse are bounded in every compact set bounded away from the hyperplane $h_d = 0$ (equivalently, $f_n = 0$). Since each component of Δf_{min} is smaller than $\|\Delta f_{min}\|$, we have

$$\begin{aligned} |h_{d,min}^m - f_n| &\le \|\Delta f_{min}\| \\ |h_{d-1,min} - F_{d-1}(h_{d,min}, f_{n-1})| &\le \|\Delta f_{min}\| \\ &\vdots \\ |h_{0,min} - F_0(h_{1,min}, \dots, h_{d,min}, f_{n-d})| &\le \|\Delta f_{min}\|. \end{aligned}$$

This relies on g being monic. By the implicit function theorem (or simple Taylor series expansion of this nonlinear function),

$$0 = F(h^{(0)}) = F(h_{min}) + J(h_{min})(h - h_{min}) + \cdots$$

Therefore $h - h_{min} = J^{-1}(-F(h_{min}))$ and $\|h^{(0)} - h_{min}\| \le K \cdot \|\Delta f_{min}\|$ since J^{-1} is bounded. We also know that $\|\Delta f_{min}\| \le \|\Delta f^{(0)}\|$, the residual of the initial approximation, and we thus have some confidence that a small residual after the first iteration ensures that $h^{(0)}$ and h_{min} will be close, provided that f_n is not too small and thus that the constant bounding J^{-1} is not too large. □

This theorem will ensure convergence of our procedures if f is near enough to a decomposable polynomial and f_n is not too small. In practice, we rely more on computation by the procedure itself than any *a priori* guarantees.

An example

Suppose that $m = 3$, $g = z^3 + g_1 z + g_0$ and $h = \varepsilon^{1/3} z^2 + z$, with $\varepsilon > 0$. Then put

$$\begin{aligned} f &= \delta z^6 + g \circ h \\ &= (\varepsilon + \delta) z^6 + 3\varepsilon^{2/3} z^5 + \cdots + g_1 z + g_0. \end{aligned}$$

For this f, the algorithm of this paper produces the following initial guess for h:

$$h^{(0)}(z) = (\varepsilon + \delta)^{1/3} z^2 + \varepsilon^{2/3} (\varepsilon + \delta)^{-2/3} z + \delta \varepsilon^{1/3} (\varepsilon + \delta)^{-5/3}. \tag{5}$$

The next step in the algorithm produces the following $g^{(0)}$:

$$g^{(0)}(z) = z^3 + c_2 z^2 + c_1 z + c_0, \tag{6}$$

with

$$\begin{aligned} c_0 &= g_0 - g_1 \varepsilon^{-4/3} \delta + O(\delta^2) \\ c_1 &= g_1 - L_1(\varepsilon) \varepsilon^{-1} \delta + O(\delta^2) \\ c_2 &= \quad - L_2(\varepsilon) \varepsilon^{-4/3} \delta + O(\delta^2) \end{aligned}$$

as $\delta \to 0$. The residual is, to $O(\delta^2)$, $f(z) - g^{(0)}\left(h^{(0)}(z)\right) =$

$$\delta \left[\frac{k_1(\varepsilon)}{\varepsilon^{1/3}} z + \frac{k_1(\varepsilon)}{\varepsilon^{2/3}} z^2 + \frac{k_3(\varepsilon)}{\varepsilon} z^3 + \frac{k_4(\varepsilon)}{\varepsilon^{2/3}} z^4 \right], \tag{7}$$

where the details of the rational functions $L_1(\varepsilon)$, $L_2(\varepsilon)$, $k_i(\varepsilon)$ are suppressed for brevity. By computation, we can bound each of these functions for all relevant ε and g_1 (the residual does not contain g_0) to find

$$\|f(z) - g^{(0)}\left(h^{(0)}(z)\right)\| \le \frac{5.3}{\varepsilon} \delta + O(\delta^2) \tag{8}$$

as $\delta \to 0$. This bound becomes larger as the leading coefficient ε of the unperturbed decomposition goes to zero in the pathological case. Theorem 3.1 shows that this behaviour is general, apart from the constant 5.3 of course.

This simultaneously exemplifies the good quality of the initial guess if $\varepsilon \gg 0$ and δ is small, and the pathology of the theorem as $\varepsilon \to 0$. Note that the effects on the coefficients of g may be larger than on the residual (specifically, $O(\varepsilon^{-4/3})$ rather than $O(\varepsilon^{-1})$). This reflects the general ill-condition of the problem of finding accurate g; recall that this ill-conditioning is a characteristic of the problem and not the algorithm.

4 LEAST-SQUARES ITERATION

The proposed algorithm breaks the problem into a sequence of alternating least-squares problems as follows:

(1) find an initial $h^{(0)}$ (as by the method of Section 3).

(2) Given $h^{(k)}$, solve a linear least-squares problem for the best possible $g^{(k)}$. Stop if the resulting $\|\Delta f\|$ is small enough or has stabilized.

(3) Given $g^{(k)}$, approximate a solution to the nonlinear least-squares problem for the best possible $h^{(k+1)}$.

Repeat steps 2 and 3 until sufficient accuracy is attained or your patience is exhausted. When this method converges, it converges linearly since it is just functional iteration (similar to that discussed in [5]).

Essentially, we ignore the interactions between the changes in h and the changes in g. By doing so, we formulate a pair of problems, together substantially smaller than the full Newton iteration studied in the next section. At each iteration we need to solve two symmetric, positive definite linear systems: a well-structured $m \times m$ system and a Toeplitz $(d+1) \times (d+1)$ system. The full Newton iteration gives rise to $(m+d+1) \times (m+d+1)$ linear systems to be solved at each iteration. This Newton iteration possibly requires fewer iterations, but each has a substantially higher cost.

4.1 Computing an optimal g given f and h.

What follows is a straightforward derivation of the normal equations defining the coefficients of g which give the minimum possible $\|\Delta f\|^2 = \|f - g \circ h\|^2$. We include this derivation here because, while it is a standard computation, it helps to clarify precisely what our proposed algorithm does.

We observe that g is simply a least-squares solution to the linear system in $g_0, \ldots, g_{m-1}$:

$$g_0 + g_1 h + g_2 h^2 + \cdots + g_{m-1} h^{m-1} = f - h^m.$$

This is easily expressed as a matrix problem. Define the $(n-d+1) \times m$ matrix $\boldsymbol{B}$ by $B_{k\ell} = [z^k]h^\ell$ for $0 \le k \le n-d$ and $0 \le \ell \le m-1$, and column vector $\boldsymbol{v}$ by $v_k = [z^k](f - h^m)$. The best possible $\boldsymbol{g}$ is obtained as a least-squares solution to $\boldsymbol{Bg} = \boldsymbol{v}$, where $\boldsymbol{g} = (g_0, \ldots, g_{m-1})^t$. The normal equations for this problem form the linear system $\boldsymbol{B}^*\boldsymbol{Bg} = \boldsymbol{B}^*\boldsymbol{v}$ for $\boldsymbol{g}$. Since the high-order coefficient of h is assumed not to be zero, $\boldsymbol{B}$ has full rank and $\boldsymbol{T} = \boldsymbol{B}^*\boldsymbol{B}$ is non-singular and positive definite.

We note that these same normal equations can also be derived via contour integrals as

$$\begin{aligned} T_{k\ell} &= \frac{1}{2\pi}\int_0^{2\pi} h^\ell(z)\overline{h^k(z)}\,dt = \frac{1}{2\pi i}\int_C h^\ell(z)\bar{h}^k(1/z)\frac{dz}{z} \\ &= [\,z^0\,]\ h^\ell(z)\bar{h}^k(1/z)\,, \end{aligned}$$

and

$$\begin{aligned} b_k &= \frac{1}{2\pi}\int_0^{2\pi} (f(z) - h^m(z))\overline{h}^k(1/z)\,dt \\ &= [\,z^0\,]\,(f(z) - h^m(z))\overline{h}^k(1/z)\,. \end{aligned}$$

The above least squares system can be constructed and solved using standard linear algebra with $O(m^2 n + m^3)$ floating point operations without relying heavily on the structure of the matrix. We can do considerably better than this as follows. Let $\omega \in \mathbb{C}$ be an $(n-d+1)$th primitive root of unity and $\boldsymbol{E} \in \mathbb{C}^{(n-d+1)\times(n-d+1)}$ the Vandermonde matrix with $E_{k\ell} = \omega^{k\ell}$. Since $\boldsymbol{E}^*\boldsymbol{E}$ is $(n-d+1)I$, the $\boldsymbol{g}$ which minimizes $\|\boldsymbol{Bg} - \boldsymbol{v}\|$ also minimizes $\|\boldsymbol{EBg} - \boldsymbol{Ev}\|$. Moreover, $\boldsymbol{V} = \boldsymbol{EB}$ is itself Vandermonde, with $V_{k\ell} = h^\ell(\omega^k)$.

Solving the least squares problem is equivalent to solving the system $\boldsymbol{Hg} = \boldsymbol{V}^t\boldsymbol{v}$ where $\boldsymbol{H} = \boldsymbol{V}^t\boldsymbol{V} \in \mathbb{C}$, an $m \times m$ Hankel matrix with $H_{kl} = \sum_{0 \le i \le n-d} h(\omega^i)^{k+\ell}$ for $0 \le k, \ell < m$. This system is non-singular and positive definite by construction. Moreover, we can compute all the power sums H_{ij} by the Newton identities and indeed solve this linear least-squares problem with $O(n \log^2 n)$ floating point operations using a super-fast Hankel solver; see Pan [11]. Finally, we note that since $\boldsymbol{g}$ is initially defined to minimize $\|\boldsymbol{Bg} - \boldsymbol{v}\|$ (and is hence the *unique* solution to a system of *real* linear equations), the obtained $\boldsymbol{g}$ lies in $\mathbb{R}^{m\times 1}$ (this despite the transformation $\boldsymbol{E}$ into $\mathbb{C}$).

In summary, each iteration of this linear least-squares step to find an optimal g given f and h can be accomplished with $O(n \log^2 n)$ floating point operations.

4.2 Computing an optimal h given f and g.

Next, we wish to compute h so that

$$\|\Delta f\|^2 = \frac{1}{2\pi}\int_0^{2\pi} \overline{(g(h) - f)}(g(h) - f)\,dt \qquad (9)$$

is minimized. This is a nonlinear least-squares problem, and we propose an iterative solution scheme to an approximate (local) minimum.

We work under the assumption that f is nearly decomposable (this is the only case that we can say anything interesting for, anyway). We also assume that the initial g and h are close to being correct, and that by linearizing about h we will not be too far from the truth. Thus

$$f(z) - g\,(h(z) + \Delta h(z)) \doteq f(z) - g(h(z)) - g'(h(z))\Delta h(z)\,,$$

ignoring quadratic terms in Δh.

So instead of trying to minimize the nonlinear equation (9), we minimize

$$\|f(z) - g(h(z)) - g'(h(z))\Delta h(z)\|^2\,. \qquad (10)$$

To solve this linear least-squares problem, let $A = f(z) - g(h(z)) - g'(h(z))\Delta h(z)$ and consider perturbations to the optimal Δh, say $\Delta h + \delta h$. We wish to choose Δh so as to minimize

$$\|A - g'(h(z))\delta h(z)\|^2\,.$$

If we can choose the coefficients of Δh so that the cross terms are zero, then as usual we have found the minimum. The cross terms are

$$\frac{1}{2\pi}\int_0^{2\pi} (\overline{A}g'(h(z))\delta h(z) + \overline{g'(h(z))\delta h(z)}A)\,dt\,. \qquad (11)$$

If this is zero for all choices of $\delta h = \sum_{\ell=0}^d \delta h_\ell z^\ell$, then

$$\begin{aligned} \|A - g'(h(z))\delta h(z)\|^2 &= \|A\|^2 + \|g'(h(z))\delta h(z)\|^2 \\ &\ge \|A\|^2\,, \end{aligned}$$

and since $g' \not\equiv 0$ we will have found the unique global minimum (though of course the global minimum of this *linear* approximation to (9) is merely 'near' to a local minimum of (9)). Necessary and sufficient conditions for this are the normal equations

$$\frac{1}{2\pi}\int_0^{2\pi} \left(f(z) - g(h(z)) - g'(h(z))\Delta h(z)\right)\overline{g}'(\overline{h}(\bar{z}))\bar{z}^k\,dt = 0\,. \qquad (12)$$

This follows on choosing $\delta h_\ell = \delta_{k\ell}$ and $\delta h_\ell = i\delta_{k\ell}$ to show that the real and imaginary parts are zero separately.

The normal equations (12) can be arranged to get

$$\sum_{\ell=0}^{d} T_{k\ell}\Delta h_\ell = b_k\,, \qquad 0 \le k \le d\,,$$

where

$$\begin{aligned} T_{k\ell} &= [z^{k-\ell}]\; g'(h(z))\,\overline{g}'(\overline{h}(1/z)) \\ b_k &= [z^k]\; (f(z) - g(h(z))\,\overline{g}'(\overline{h}(1/z))\,. \end{aligned}$$

This derivation allows for a very fast computation of the entries in $\boldsymbol{T}$ through series manipulation. To solve stably and efficiently such a system it is also necessary to know that it is non-singular and positive definite as well as Hermitian and Toeplitz. To see this, we observe that $\boldsymbol{T}$ factors as $\boldsymbol{T} = \boldsymbol{B}^*\boldsymbol{B}$, where $\boldsymbol{B}$ is an $(n+d) \times (d+1)$ matrix with $B_{k\ell} = [z^k]z^\ell g'(h(z))$. This is a lower triangular Toeplitz matrix of full column rank when $g'(h(z))$ is non-zero, whence $\boldsymbol{B}^*\boldsymbol{B}$ is non-singular and positive definite.

Once we have computed a Δh using this linear least-squares formulation we may then update $h := h + \Delta h$. The entire process can then be iterated by re-linearizing around this new h and again approximating a Δh minimizing $\|f - g(h+\Delta h)\|$. Since this nonlinear least-squares problem is only a component in the entire solver it is not clear that it is necessary to repeat this local process (for fixed f and g) until convergence occurs. However, we have seen examples in which substantial convergence is required in this sub-problem for a globally minimal decomposition to be obtained.

Computationally, each iteration of this nonlinear least-squares solver has very low cost. Each system $\boldsymbol{T}$ can be easily constructed using only series manipulations. $\boldsymbol{T}$ can be constructed with $O(n \log^2 n)$ operations using the series manipulation algorithms of Brent & Kung [4] and an FFT for polynomial multiplication. The solution to the system can be obtained via the stable Toeplitz solvers of Trench [14] using (d^2) operations, or the fast and stable methods (for positive definite matrices) [3, 12] which require $O(d \log^2 d)$ operations. In summary, each iteration requires $O(n \log^2 n)$ floating point operations.

5 NEWTON ITERATION

In this section we explore the direct use of Newton's method to solve the nonlinear minimization problem: find $g, h \in \mathbb{R}[z]$ minimizing $\|g \circ h - f\|^2$. We give an effective method of computing the requisite derivatives analytically, and implement, test, and compare the method with the sequence of linear least-squares problems of the earlier section.

We consider

$$N_f(g+\Delta g, h+\Delta h) = \|f - (g+\Delta g)\circ(h+\Delta h)\|^2$$

with

$$\Delta h(z) = \sum_{\ell=0}^{d} \Delta h_\ell z^\ell, \quad \Delta g(z) = \sum_{\ell=0}^{m-1} \Delta g_\ell z^\ell.$$

Assume for the purpose of exposition that f, g, h, Δf, Δg, and Δh are all in $\mathbb{R}[z]$; however $z \in \mathbb{C}$. Lemma 2.1 is used to compute N_f. Denoting the integrand of the integral for N_f by I_f, we expand to second order in Δg, Δh.

$$\begin{aligned} I_f(g+\Delta g, h+\Delta h) = &(g(h(z)) - f(z))(g(h(\bar z)) - f(\bar z)) \\ &+ g'(h(\bar z))(g(h(z)) - f(z))\Delta h(\bar z) \\ &+ (g(h(z)) - f(z))\Delta g(h(\bar z)) \\ &+ g'(h(z))g'(h(\bar z))\Delta h(z)\Delta h(\bar z) \\ &+ (\Delta g(h(z))\Delta g(h(\bar z))) \\ &+ g'(h(\bar z))\Delta g(h(z))\Delta h(\bar z) \\ &+ (g(h(z)) - f(z))\Delta g'(h(\bar z))\Delta h(z) \\ &+ \frac{1}{2} g''(h(\bar z))(g(h(z)) - f(z))\Delta h^2(\bar z) \\ &+ \text{c. c.}, \end{aligned}$$

where c. c. indicates the complex conjugate of all non-real summands.

We write $\boldsymbol{x}^t = [\boldsymbol{\Delta g}^t, \boldsymbol{\Delta h}^t]$, where $\boldsymbol{\Delta h}^t$ and $\boldsymbol{\Delta g}^t$ indicate the vectors of coefficients

$$\begin{aligned} \boldsymbol{\Delta h}^t &= [\Delta h_0, \Delta h_1, \ldots, \Delta h_d], \\ \boldsymbol{\Delta g}^t &= [\Delta g_0, \Delta g_1, \ldots, \Delta g_{m-1}]\,. \end{aligned}$$

(Note that $g + \Delta g$ is monic and hence $\Delta g_m = 0$). We have the following best quadratic approximation to N_f.

$$N_f(g+\Delta g, h+\Delta h) = N_f(g,h) + \boldsymbol{b}^t\boldsymbol{x} + \boldsymbol{x}^t\boldsymbol{A}\boldsymbol{x} + \cdots \quad (13)$$

for the following $1 \times (d+1+m)$ vector $\boldsymbol{b}^t$ and $(d+1+m) \times (d+1+m)$ symmetric matrix $\boldsymbol{A}$:

$$\begin{aligned} \boldsymbol{b}^t &= \nabla_x N_f(g,h) \\ \boldsymbol{A} &= \frac{1}{2}\nabla_x\nabla_x N_f(g,h) \end{aligned} \quad (14)$$

The key technique for identification of these matrices and vectors from the series computation (13) is the writing of polynomial products as vector-matrix products:

$$\begin{aligned} (\Delta h)^2 &= \Delta h \cdot \Delta h \\ &= [\Delta h_0, \Delta h_1, \ldots \Delta h_d] \begin{bmatrix} 1 \\ z \\ \vdots \\ z^d \end{bmatrix} [1, z, \ldots, z^d] \begin{bmatrix} \Delta h_0 \\ \Delta h_1 \\ \vdots \\ \Delta h_d \end{bmatrix} \end{aligned}$$

By association, we group the rank-1 product in the middle to identify the desired matrix block.

We then apply the $\frac{1}{2\pi}\int_0^{2\pi} dt$ operator, from Lemma 2.1, which simply computes the constant coefficient in the series expansion.

Solving for x in a Newton step

The right-hand side of (13) is minimized using an eigenvalue factoring. We compute an eigenvalue factoring rather than the cheaper LDL^t factoring, because the condition number of A was observed experimentally to be exponential in $m+d$. However, the larger eigenvalues of $\boldsymbol{A}$ are well-conditioned because $\boldsymbol{A}$ is symmetric [6]: indeed, if $\boldsymbol{A}$ is perturbed to $\boldsymbol{A} + \varepsilon\boldsymbol{E}$, its eigenvalues change from λ to $\lambda + \varepsilon(\boldsymbol{q}^t\boldsymbol{E}\boldsymbol{q}) + O(\varepsilon^2)$, where $\boldsymbol{q}$ is a normalized eigenvector associated with λ. That is, in an absolute sense, the uncertainties in the eigenvalues are no larger than the uncertainties in $\boldsymbol{A}$. The small eigenvalues, of course, may be greatly perturbed in a relative sense, but, as we will see below, to stabilize the Newton step we will ignore eigenvalues that are too small.

We write $\boldsymbol{A} = \boldsymbol{Q\Lambda Q}^t$, where $\boldsymbol{\Lambda}$ is the usual diagonal matrix of eigenvalues and $\boldsymbol{Q}$ is orthonormal. Now substitute $\boldsymbol{x} = \boldsymbol{Qy}$ in (13) to get

$$\begin{aligned} N_f + \boldsymbol{b}^t\boldsymbol{x} + \boldsymbol{x}^t\boldsymbol{A}\boldsymbol{x} =& N_f + \boldsymbol{b}^t\boldsymbol{Q}\boldsymbol{y} + \boldsymbol{y}^t\boldsymbol{Q}^t\boldsymbol{A}\boldsymbol{Q}\boldsymbol{y} \\ =& \boldsymbol{b}^t\boldsymbol{Q}\boldsymbol{y} + \boldsymbol{y}^t\boldsymbol{\Lambda}\boldsymbol{y}. \end{aligned}$$

The constant N_f can be dropped. Denoting $\boldsymbol{b}^t\boldsymbol{Q} = -2\boldsymbol{p}^t$, we have

$$\begin{aligned} \boldsymbol{b}^t\boldsymbol{Q}\boldsymbol{y} + \boldsymbol{y}^t\boldsymbol{\Lambda}\boldsymbol{y} =& -2p_1y_1 - 2p_2y_2 - \cdots - 2p_my_m \\ &+ \lambda_1y_1^2 + \lambda_2y_2^2 + \cdots + \lambda_my_m^2. \end{aligned}$$

If any $\lambda_k < 0$, then there is no local minimum nearby. Since $\lambda_k y_k^2 - 2p_ky_k = \lambda_k(y_k - p_k/\lambda_k)^2 - p_k^2/\lambda_k$, for $\lambda_k \neq 0$, the minimum of the quadratic form is $y_k = p_k/\lambda_k$. Since $\boldsymbol{A}$ is symmetric, the factoring $A = \boldsymbol{Q}^t\boldsymbol{\Lambda}\boldsymbol{Q}$ is effectively the same as the SVD. By ignoring eigenvalues with magnitude smaller than some useful cutoff $\varepsilon\|\boldsymbol{\Lambda}\|_\infty$ and preventing large y_k solutions (which would produce large $\boldsymbol{x}$ terms and therefore invalidate the assumption under which we linearized the problem) we therefore stabilize the Newton step.

6 EXPERIMENTAL IMPLEMENTATION

Below we give examples from experimental implementations of both methods. The same function f is used for each method. As will be seen, Newton's method, given first, converges rapidly. In contrast, the iterated least-squares method takes more iterations (as expected), showing linear convergence. Each iteration is substantially simpler and cheaper, though, and the two methods seem comparable in overall efficiency. The iterated least-squares method appears to be faster on larger problems, for low accuracy requirements.

Both methods can fail in pathological cases with the leading coefficients of f being 'too small'. In the pathological cases that we have tried, both methods converge to the same incorrect local minimum.

Input polynomial. This input polynomial and precision is chosen to show the typical features of the two methods: that the Newton iteration converges in fewer iterations than the linearly convergent iterated least-squares method, and, moreover, the initial guess is usually so good that relatively few iterations are needed by either method.

$$\begin{aligned} f :=\ & 0.01928725741\, z^{20} + 0.01542980592\, z^{19} \\ &+ 0.004628941775\, z^{18} + 0.0006171922369\, z^{17} \\ &+ 0.07717988923\, z^{16} + 0.1620129623\, z^{15} \\ &+ 0.07869201021\, z^{14} + 0.01450401757\, z^{13} \\ &+ 0.1171492929\, z^{12} + 0.3934600510\, z^{11} \\ &+ 0.3749442839\, z^{10} + 0.1064656608\, z^{9} \\ &+ 0.08640691316\, z^{8} + 0.3626004392\, z^{7} \\ &+ 0.5323283043\, z^{6} + 0.3047386670\, z^{5} \\ &+ 0.06171922370\, z^{4} + 0.1157235444\, z^{3} \\ &+ 0.2314470889\, z^{2} + 0.2122598235\, z \\ &+ 0.09643628703 \end{aligned}$$

Initial approximations

$$\begin{aligned} g =\ & 0.006244011759 + 0.1017654369\, z \\ &- 0.2037319773\, z^2 - 0.005002615065\, z^3 \\ &+ 1.002041909\, z^4 \end{aligned}$$

$$\begin{aligned} h =\ & 0.3726641527\, z^5 + 0.07453283050\, z^4 \\ &+ 0.7453283054\, 10^{-11}\, z^3 \\ &- 0.3726641527\, 10^{-11}\, z^2 + 0.3726641527\, z \\ &+ 0.5589962294 \end{aligned}$$

$$||f - g \circ h|| = 0.0003707618881$$

Newton Iteration 1

$$\begin{aligned} g =\ & 0.006307674448 + 0.1012707551\, z \\ &- 0.2025224484\, z^2 - 0.005113811181\, z^3 \\ &+ 1.002041909\, z^4 \end{aligned}$$

$$\begin{aligned} h =\ & 0.3726154584\, z^5 + 0.07467910016\, z^4 \\ &- 0.0001353050967\, z^3 \\ &+ 0.00005400150629\, z^2 + 0.3727592640\, z \\ &+ 0.5587404203 \end{aligned}$$

$$||f - g \circ h|| = 0.0003372925949$$

Newton Iteration 2

$$\begin{aligned} g =\ & 0.006307955886 + 0.1012688703\, z \\ &- 0.2025174571\, z^2 - 0.005114263058\, z^3 \\ &+ 1.002041909\, z^4 \end{aligned}$$

$$\begin{aligned} h =\ & 0.3726153448\, z^5 + 0.07467965426\, z^4 \\ &- 0.0001357186198\, z^3 \\ &+ 0.00005416053562\, z^2 + 0.3727596011\, z \\ &+ 0.5587393242 \end{aligned}$$

$$||f - g \circ h|| = 0.0003372922955$$

Least Squares Iteration 1

$$\begin{aligned} g =\ & 0.00625752328 + 0.1015815394\, z \\ &- 0.2032108868\, z^2 - 0.005551305342\, z^3 \\ &+ 1.002248347\, z^4 \end{aligned}$$

$$\begin{aligned} h =\ & 0.3725977213\, z^5 + 0.07463618893\, z^4 \\ &- 0.0001369138656\, z^3 \\ &+ 0.00009730067631\, z^2 + 0.3726503289\, z \\ &+ 0.5589999916 \end{aligned}$$

$$||\, f - g \circ h\, || = 0.0003406441482$$

Least Squares Iteration 2

$$g = 0.006265045950 + 0.1015143147\,z - 0.2029731112\,z^2 - 0.005824248119\,z^3 + 1.002355603\,z^4$$

$$h = 0.3725872261\,z^5 + 0.07464906634\,z^4 - 0.0001381228199\,z^3 + 0.00008297688615\,z^2 + 0.3726721048\,z + 0.5589876281$$

$$||\,f - g \circ h\,|| = 0.0003386563417$$

Least Squares Iteration 3

$$g = 0.006269758096 + 0.1014732862\,z - 0.2028247851\,z^2 - 0.0059956957\,z^3 + 1.002423191\,z^4$$

$$h = 0.3725805736\,z^5 + 0.07465676477\,z^4 - 0.0001372994316\,z^3 + 0.00007259903511\,z^2 + 0.3726868766\,z + 0.5589792377$$

$$||\,f - g \circ h\,|| = 0.0003378490878$$

Least Squares Iteration 4

$$g = 0.006272764905 + 0.1014471517\,z - 0.2027302025\,z^2 - 0.006105053024\,z^3 + 1.002466306\,z^4$$

$$h = 0.3725763263\,z^5 + 0.07466166595\,z^4 - 0.0001367225764\,z^3 + 0.00006593375947\,z^2 + 0.3726963367\,z + 0.5589738631$$

$$||\,f - g \circ h|| = 0.0003375194860$$

7 CONCLUDING REMARKS

This paper has established a framework for the decomposition of approximate polynomials. We have explored two algorithms for computing such decompositions. The following interesting questions suggest themselves for future study.

1. Are the iterative algorithms that we have presented numerically stable? All indications suggest that they are, except near the described pathological cases.
2. Can one find *a priori* bounds for the basins of attraction of each of the proposed methods?
3. Can the structure of the Jacobian in the full Newton iteration of Section 5 be exploited to yield faster iterations?
4. What can be said about the distance to the nearest decomposable polynomial from any given one?

References

[1] C. Alonso, J. Gutiérrez, and T. Recio. A rational function decomposition algorithm by near-separated polynomials. *J. Symb. Comp*, 19(6):527–544, 1995.

[2] D. R. Barton and R. Zippel. Polynomial decomposition algorithms. *J. Symb. Comp.*, 1:159–168, 1985.

[3] R. R. Bitmead and B. D. O. Anderson. Asymptotically fast solution of Toeplitz and related systems of equations. *Linear Algebra and Appl.*, 34:103–116, 1980.

[4] R. P. Brent and H. T. Kung. Fast algorithms for manipulating formal power series. *J. Assoc. Comput. Mach.*, 25:581–595, 1978.

[5] P. Chin, R. Corless, and G. Corliss. Optimization strategies for the approximate gcd problem. In *Proc. ISSAC'98*, pages 228–235, Maui, Hawaii, 1998. ACM Press.

[6] Gene H. Golub and Charles Van Loan. *Matrix Computations.* Johns Hopkins, 3rd edition, 1995.

[7] R. Graham, D. Knuth, and O. Patashnik. *Concrete Mathematics.* Addison-Wesley, Reading, MA, USA, 1994.

[8] J. Gutiérrez, T. Recio, and C. Ruiz de Velasco. Polynomial decomposition algorithm of almost quadratic complexity. In *Proc. Applied algebra, algebraic algorithms and error-correcting codes*, volume 357 of *Lecture Notes in Comp. Sci.*, pages 471–475, Rome, 1988. Springer, Berlin.

[9] Thomas W. Körner. *Fourier Analysis.* Cambridge University Press, 1988.

[10] D. Kozen and S. Landau. Polynomial decomposition algorithms. *J. Symb. Comp.*, 7:445–456, 1989.

[11] V. Pan. Parallel least-squares solution of general and Toeplitz-like linear systems. In *Proc. 2nd Annual ACM Symp. on Parallel Algorithms and Architecture*, pages 244–253, 1990.

[12] V. Pan and R. Schreiber. A fast, preconditioned conjugate gradient Toeplitz solver. *Comp. Math. Appl.*, 24(7):17–24, 1992.

[13] T. Rivlin. *Chebyshev polynomials: from approximation theory to number theory.* Wiley, 1990.

[14] W. Trench. An algorithm for the inversion of finite Toeplitz matrices. *SIAM J. Applied Mathematics*, 12:515–522, 1964.

[15] J. von zur Gathen. Functional decomposition of polynomials: the tame case. *J. Symb. Comp.*, 9:281–299, 1990.

[16] R. Zippel. Rational function decomposition. In *Proc. ISSAC'91*, pages 1–6. ACM Press, 1991.