Symbolic-Numeric Sparse Interpolation of Multivariate Polynomials

[Extended Abstract]

Mark Giesbrecht and George Labahn
School of Computer Science, University of Waterloo
Waterloo, Ontario N2L 3G1, Canada

and
Wen-shin Lee
INRIA, GALAAD
BP 93, 06902 Sophia-Antipolis, France

1 Introduction

In many applications, multivariate polynomials are encountered in such a way that an implicit representation is the most cost effective for computation. A black box representation of a multivariate polynomial is a procedure such that, for any given input, it outputs the evaluation of the polynomial at that input. Black box polynomials appear naturally in applications such as multivariate polynomial systems (Corless et al. 2001, both approximate and exact) and the manipulation of sparse polynomials, such as factoring (Kaltofen and Trager 1990; Díaz and Kaltofen 1998).

We consider the problem of sparse interpolation of a multivariate polynomial given by a floating-point black box. That is, both the inputs and outputs of the black box are precise up to a fixed number of digits. As a result, the coefficients in the target polynomial can only be known up to a certain precision.

For example, a floating point black box can be a procedure that, for any given input, computes the value of the determinant of a matrix of multivariate polynomials with floating point coefficients. Such a black box evaluation procedure could be constructed from a number of effective numeric algorithms for finding a determinant, for example Gaussian elimination.

A typical problem with black box representations is to convert such objects into a standard representation. That is, given a multivariate polynomial $f(x_1, \ldots, x_n)$ in black box form, we want to find powers $(d_{j_1}, \ldots, d_{j_n})$ and non-zero coefficients c_j such that

$$f(x_1, \cdots, x_n) = \sum_{j=1}^{t} c_j x_1^{d_{j1}} \cdots x_n^{d_{jn}}.$$

In the case of multivariate polynomials, one often expects such a representation to be sparse. The best known interpolation methods that are sensitive to the sparsity of the target polynomial are the Ben-Or/Tiwari algorithm (Ben-Or and Tiwari 1988) and Zippel's method (Zippel 1979). Although both approaches have been generalized and improved (Zippel 1990; Kaltofen and Lakshman Yagati 1988; Grigoriev et al. 1990; Zilic and Radecka 1999), they all require exact arithmetic. In quite a different model, Mansour (Mansour 1995) gives a nonetheless impressive randomized algorithm for interpolating a sparse integer polynomial from (limited precision) interpolation points. While the algorithm guarantees an answer with controllably high probability, its cost is quite dependent on the size L (regardless of its precision) of the largest coefficient in f, as well as the sparsity t and degree: it requires about $O((\log L)^8 t \log \deg f)$ bit operations.

In numeric arithmetic, a procedure comparable to Ben-Or/Tiwari algorithm dates back to Baron de Prony in 1795 (Prony III (1795); Brezinski 1991), which actually considers the interpolation problem of fitting a sum of univariate exponential functions

$$F(x) = \sum_{j=1}^{t} c_j e^{\mu_j x}.$$

Prony's method determines c_j and μ_j from the evaluations of $F(x)$ at equally spaced points $F(0), F(1), \ldots$. There are many interesting variations of this numeric interpolation problem, for example, the problem of shape from moments (Milanfar et al. 1995; Golub et al. 1999). Some other examples are tomography (Milanfar et al. 1995) and signal decomposition (Marple, Jr. 1987).

We develop sparse interpolation algorithms for multivariate polynomials in floating point arithmetic. We also take advantage of the current state of the art algorithms and look at the numerical accuracy for numeric subproblems. Finally, we conclude with a discussion of relavant topics and future research.

2 Preliminaries

In this section we describe the Prony's method for interpolating sums of exponentials and the Ben-Or/Tiwari algorithm for multivariate polynomials. We show that these two algorithms are closely related.

2.1 Prony's method

Prony's method (Prony III (1795)) seeks to interpolate a univariate $F(x)$ that is a sum of exponential functions. That is, it tries to determine c_j and μ_j such that

$$F(x) = \sum_{j=1}^{t} c_j e^{\mu_j x} \text{ with } c_j \neq 0.$$

Since there are $2t$ unknowns, one would expect a system of at least the same number of equations. However, these equations are not linear. Prony's method converts the nonlinear component to the root finding of a single, univariate polynomial. All other steps involve the solving systems of (structured) linear equations.

Let $b_j = e^{\mu_j}$, then $F(x) = \sum_{j=1}^{t} c_j e^{\mu_j x} = \sum_{j=1}^{t} c_j b_j^x$. Consider the polynomial $\Lambda_F(z)$ having the b_j's as zeros:

$$\Lambda_F(z) = \prod_{j=1}^{t} (z - b_j) = z^t + \lambda_{t-1} z^{t-1} + \cdots \lambda_1 z + \lambda_0.$$

Here are the key facts used in the algorithm: the sequence $F(0), F(1), F(2), \ldots$ is linearly generated, and its minimal generating polynomial is Λ_F (Hildebrand 1956; Ben-Or and Tiwari 1988).

The polynomial $\Lambda_F(z)$ can be determined by solving a Hankel system:

$$\begin{bmatrix} F(0) & F(1) & \ldots & F(t-1) \\ F(1) & F(2) & \ldots & F(t) \\ \vdots & \vdots & \ddots & \vdots \\ F(t-1) & F(t) & \ldots & F(2t-2) \end{bmatrix} \begin{bmatrix} \lambda_0 \\ \vdots \\ \lambda_{t-2} \\ \lambda_{t-1} \end{bmatrix} = - \begin{bmatrix} F(t) \\ F(t+1) \\ \vdots \\ F(2t-1) \end{bmatrix}, \tag{1}$$

which is also called the *Yule-Walker* equations for the series $\sum_{j \geq 0} F(j) z^j$.

Finding zeros for Λ_F can determine $b_1, \ldots, b_t$ (thus $\mu_1, \ldots, \mu_t$). The coefficients $c_1, \ldots, c_t$ can be computed by solving a transposed Vandermonde system:

$$\begin{bmatrix} 1 & \cdots & 1 \\ b_1 & \cdots & b_t \\ \vdots & \ddots & \vdots \\ b_1^{t-1} & \cdots & b_t^{t-1} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_t \end{bmatrix} = \begin{bmatrix} F(0) \\ F(1) \\ \vdots \\ F(t-1) \end{bmatrix} \tag{2}$$

2.2 The Ben-Or/Tiwari method

For a given black box polynomial f with n variables, the Ben-Or/Tiwari method (Ben-Or and Tiwari 1988) finds coefficients c_j and integer exponents $(d_{j_1}, \ldots, d_{j_n})$ such that

$$f(x_1, \ldots, x_n) = \sum_{j=1}^{t} c_j x_1^{d_{j_1}} \cdots x_n^{d_{j_n}} \text{ with } c_j \neq 0.$$

Let $\beta_j(x_1, \ldots, x_n) = x_1^{d_{j_1}} \cdots x_n^{d_{j_n}}$ be the j-th term in f, and set $b_j = \beta_j(p_1, \ldots, p_n) = p_1^{d_{j_1}} \cdots p_n^{d_{j_n}}$ with $p_1, \ldots, p_n$ pairwise relatively prime. Note that $b_j^k = \beta_j(p_1^k, \ldots, p_n^k)$ for any power k.

Then we consider the function $F(k) = f(p_1^k, \ldots, p_n^k)$. Designed for exact arithmetic, the Ben-Or/Tiwari algorithm solves Yule-Walker equations in (1) by the Berlekamp/Massey algorithm from coding theory. Once the terms b_j are found through the root finding of $\Lambda_F(z) = 0$, the exponents $(d_{j_1}, \ldots, d_{j_n})$ can be determined via repeatedly dividing b_j by $p_1, \ldots, p_n$ that are relatively prime. Finally, the coefficients c_j are determined via solving a Vandermonde system similar to (2).

3 Numerical methods in sparse interpolations

We give two sparse algorithms for interpolating black box multivariate polynomials in floating point arithmetic. One mainly follows the steps of the Ben-Or/Tiwari algorithm (Ben-Or and Tiwari 1988), while the other takes advantage of the generalized eigenvalue reformulation of Prony's method (Golub et al. 1999). We also look at the stability of the main subproblems in these algorithms.

3.1 Ben-Or/Tiwari algorithm in floating point arithmetic

If the steps of the Ben-Or/Tiwari algorithm are directly executed in floating point arithmetic, severe difficulties arise at varying stages of the computation. First, to solve the Yule-Walker equations in (1), rather than using the Berlekamp/Massey algorithm in exact arithmetic, we need to solve a Hankel system that is often ill-conditioned, especially in the case of real numbers (Beckermann 2000). Even worse, the solutions λ_j to such Hankel system are coefficients of the generating polynomial $\Lambda(z) = z^t + \lambda_{t-1} z^{t-1} + \cdots + \lambda_1 z + \lambda_0$ for root finding, while root finding is usually very sensitive to the perturbations in λ_j. Finally, the coefficients c_j of our target polynomial are determined via solving a Vandermonde system, which might be ill-conditioned.

The above difficulties are also shared by Prony's method for interpolating a sum of exponential functions, even though it was designed and used as a numerical method. Special challenges also exist for multivariate polynomial interpolations. For example, unlike the case in exact arithmetic, we can no longer use integer factorizations to recover the exponent of each variable in a multivariate term.

By evaluating each variable at powers of an appropriate primitive root of unity, we can improve the conditioning of the associated Hankel system, the computation of zeros for the generating polynomial, and the Vandermonde system. Furthermore, the exponent of each variable in a multivariate term can also be recovered.

Our strategy is to evaluate at powers of primitive roots of unity whose orders are pairwise relatively prime. Let $f(x_1, \ldots, x_n) = \sum_{j=1}^{t} c_j x_1^{d_{j_1}} \cdots x_n^{d_{j_n}} = \sum_{j=1}^{t} c_j \beta_j(x_1, \ldots, x_n)$ with $c_j \neq 0$. If $p_1, \ldots, p_n \in \mathbb{Z}_{>0}$ are pairwise relatively prime and $p_k > \deg_{x_k} f$ for $1 \leq k \leq n$ (assume we have an upper degree bound for every variable in f). Consider the following sequence for interpolation:

$$\alpha_s = f(\omega_1^s, \omega_2^s, \ldots, \omega_n^s) \text{ for } 0 \le s \le 2t-1$$

with $\omega_k = \exp(2\pi i/p_k)$. Set $m = p_1 \cdots p_n$ and $\omega = \exp(2\pi i/m)$, then $\omega_k = \omega^{m/p_k}$ for $1 \le k \le n$.

In $f(\omega_1, \ldots, \omega_n)$, each term $\beta_j(x_1, \ldots, x_n)$ is mapped to value $\beta_j(\omega_1, \ldots, \omega_n) = \omega^{d_j}$. In a numeric setting, each d_j can be computed by rounding $\log_\omega(\beta_j(\omega_1, \ldots, \omega_n))$ to the nearest integer. Then the exponent for each variable $(d_{j_1}, \ldots, d_{j_n}) \in \mathbb{Z}^n_{>0}$ can be uniquely determined by the reverse steps of the Chinese remainder algorithm (cf. Geddes et al. (1992)). That is, $d_j \bmod p_k \equiv d_{j_k}$ for $1 \le k \le n$, and[1]

$$d_j = d_{j_1} \cdot \left(\frac{m}{p_1}\right) + \cdots + d_{j_n} \cdot \left(\frac{m}{p_n}\right). \tag{3}$$

In the remaining of this subsection, we look at the numerical sensitivity for solving the associated Hankel system and the root finding of generating polynomial $\Lambda(z)$. The separation of powers for polynomial terms and the solving of the associated Vandermonde system will be addressed in subsections 3.3 and 3.4.

Solving the associated Hankel system

Consider polynomial $f(x_1, \ldots, x_n) = \sum_{j=1}^t c_j x_1^{d_{j_1}} \cdots x_n^{d_{j_n}} = \sum_{j=1}^t c_j \beta_j(x_1, \ldots, x_n)$ and the evaluation sequence $\alpha_s = f(a_1^s, \ldots, a_n^s)$ for $0 \le s \le 2t-1$. We need to solve the following Hankel systems $H_{0,t-1}$:

$$\underbrace{\begin{bmatrix} \alpha_0 & \alpha_1 & \ldots & \alpha_{t-1} \\ \alpha_1 & \alpha_2 & \ldots & \alpha_t \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{t-1} & \alpha_t & \ldots & \alpha_{2t-2} \end{bmatrix}}_{H_{0,t-1}} \begin{bmatrix} \lambda_0 \\ \lambda_1 \\ \vdots \\ \lambda_{t-1} \end{bmatrix} = - \begin{bmatrix} \alpha_t \\ \alpha_{t+1} \\ \vdots \\ \lambda_{2t-1} \end{bmatrix}$$

In general, if a polynomial f is evaluated at powers of a real value, the difference between the scale of varying powers contributes to the ill conditioning of the Hankel system. This problem is avoided in our method, since our $H_{0,t-1}$ is formed from the evaluations on a unit circle.

Now let $b_j = \beta_j(\omega_1, \ldots, \omega_n)$, $D = \mathrm{diag}(c_1, \ldots, c_t)$, and

$$V = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ b_1 & b_2 & \ldots & b_t \\ \vdots & \vdots & \ddots & \vdots \\ b_1^{t-1} & b_2^{t-1} & \ldots & b_t^{t-1} \end{bmatrix}.$$

The Hankel system $H_{0,t-1}$ can be factorized as $H_{0,t-1} = VDV^{\mathrm{Tr}}$. We make use of this factorization for investigating the condition of $H_{0,t-1}$.

[1] Recall that in exact arithmetic, the original Ben-Or/Tiwari algorithm evaluates variables at values that are pairwise relatively prime.

Lemma 1

$$\|H_{0,t-1}^{-1}\| \geq \frac{1}{t} \max_j \frac{1}{|c_j|}.$$

Proof. Let D_j be the matrix derived from D by using 0 to replace c_j in the diagonal, then matrix $V D_j V^{\mathrm{Tr}}$ is singular for $1 \leq j \leq t$.

Based on the Eckart-Young theorem and that b_j's are on the unit circle,

$$\begin{aligned}\frac{1}{\|H_{0,t-1}^{-1}\|} &= \min\{\|H_{0,t-1} - A\|,\ A \text{ singular }\} \\ &\leq \min\{\|H_{0,t-1} - V D_j V^{\mathrm{Tr}}\|\} \\ &\leq \|[1, b_j, \ldots, b_j^{t-1}]\|^2 \cdot |c_j| \leq t \cdot |c_j|. \quad \square\end{aligned}$$

Lemma 2

$$\|H_{0,t-1}^{-1}\| \leq \|V^{-1}\|^2 \cdot t \cdot \max_j \frac{1}{|c_j|}.$$

Proof. Consider $H_{0,t-1}^{-1} = (V^{\mathrm{Tr}})^{-1} D^{-1} V^{-1}$, then

$$\begin{aligned}\|H_{0,t-1}^{-1}\| &\leq \|V^{-1}\|^2 \|D^{-1}\| \leq \|V^{-1}\|^2 \cdot \sum_{j=1}^{t} \|D^{-1} e_j\| \\ &\leq \|V^{-1}\|^2 \cdot t \cdot \max_j \frac{1}{|c_j|}. \quad \square\end{aligned}$$

An upper bound for $\|H_{0,t-1}^{-1}\|$ involves the conditioning of the Vandermonde system V, which will be discussed in subsection 3.4.

Root finding of the generating polynomial

The solutions of the associated Hankel system λ_j are coefficients in polynomial $\Lambda(z) = z^t + \lambda_{t-1} z^{t-1} + \cdots + \lambda_1 z + \lambda_0$. In our floating point Ben-Or/Tiwari interpolation steps, zeros of $\Lambda(z)$ are $b_j = \beta_j(\omega_1, \ldots, \omega_n)$, where $f(x_1, \ldots, x_n) = \sum_{j=1}^{t} c_j x_1^{d_{j_1}} \cdots x_n^{d_{j_n}} = \sum_{j=1}^{t} c_j \beta_j(x_1, \ldots, x_n)$, and b_j are on the unit circle.

It is well known that root finding for a polynomial is generally poorly conditioned with respect to perturbations in the coefficients (Wilkinson 1963). However, the conditioning is greatly improved when all the roots are on the unit circle.

Let b_k be a zero of $\Lambda(z)$ and $\tilde{b}_k$ a zero of $\Lambda(z) + \epsilon\Gamma(z)$, where

$$\Gamma(z) = \gamma_t z^t + \gamma_{t-1} z^{t-1} + \cdots \gamma_0,$$

then $\tilde{b}_k \approx b_k + \zeta_1 \epsilon$ and $\Lambda(b_k + \zeta_1\epsilon) + \epsilon\Gamma(b_k + \zeta_1\epsilon) \approx 0$. By the Taylor's expansion with respect to b_k,

$$\sum_{j=0}^{t} \frac{1}{j!} \Lambda^{(j)}(b_k) \cdot (\zeta_1 \epsilon)^j + \epsilon \sum_{j=0}^{t} \frac{1}{j!} \Gamma^{(j)}(b_k) \cdot (\zeta_1 \epsilon)^j \approx 0.$$

Recall that $\Lambda(b_k) = 0$, and consider only the first order terms in ϵ, we have $\Lambda^{(1)}(b_k) \cdot \zeta_1 \epsilon + \epsilon \Gamma(b_k) \approx 0$ and

$$|\zeta_1| \approx \left| \frac{\Gamma(b_k)}{\Lambda^{(1)}(b_k)} \right| \leq \frac{\sum_{j=0}^{t} |\gamma_j|}{|\prod_{j \neq k}(b_k - b_j)|}.$$

Therefore,

$$|b_k - \tilde{b}_k| < \epsilon \cdot \frac{K_1}{|\prod_{j \neq k}(b_k - b_j)|} + K_2 \epsilon^2.$$

The size of $|\prod_{j \neq k}(b_k - b_j)|$ depends on the distribution of b_j's on the unit circle (cf. subsection 3.4).

3.2 Generalized eigenvalue reformulation

We present another interpolation algorithm by adapting the reformulation of Prony's method that combines the solving of a Hankel system and the root finding of a polynomial into a single generalized eigenvalue problem (Golub et al. 1999). As a result, both solving the Hankel system and finding roots for a polynomial can be avoided.

Consider polynomial $f(x_1, \ldots, x_n) = \sum_{j=1}^{t} c_j x_1^{d_{j_1}} \cdots x_n^{d_{j_n}} = \sum_{j=1}^{t} c_j \beta_j(x_1, \ldots, x_n)$ and the evaluation sequence $\alpha_s = f(a_1^s, \ldots, a_n^s)$ for $0 \leq s \leq 2t-1$. Define Hankel systems

$$H_{0,t-1} = \begin{bmatrix} \alpha_0 & \ldots & \alpha_{t-1} \\ \vdots & \ddots & \vdots \\ \alpha_{t-1} & \ldots & \alpha_{2t-2} \end{bmatrix} \text{ and } H_{1,t} = \begin{bmatrix} \alpha_1 & \ldots & \alpha_t \\ \vdots & \ddots & \vdots \\ \alpha_t & \ldots & \alpha_{2t-1} \end{bmatrix}$$

so that $H_{1,t}$ is one row shifted up from $H_{0,t-1}$. Let $b_j = \beta_j(a_1, \ldots, a_n)$. If we set $Z = \text{diag}(b_1, \ldots, b_t)$, $D = \text{diag}(c_1, \ldots, c_t)$, and

$$V = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ b_1 & b_2 & \ldots & b_t \\ \vdots & \vdots & \ddots & \vdots \\ b_1^{t-1} & b_2^{t-1} & \ldots & b_t^{t-1} \end{bmatrix},$$

then $H_{0,t-1} = VDV^{\text{Tr}}$, $H_{1,t} = VDZV^{\text{Tr}}$. The solutions λ to the generalized eigenvalue problem

$$H_{1,t} v = \lambda H_{0,t-1} v \tag{4}$$

are $b_1, \ldots, b_t$, the terms $\beta_j(x_1, \ldots, x_n)$ evaluated at $(a_1, \ldots, a_n)$. If $a_1, \ldots, a_n$ are chosen as $\omega_1, \ldots, \omega_n$ in subsection 3.1, the multivariate terms $\beta_j(x_1, \ldots, x_n)$ can be recovered accordingly.

Golub, Milanfar and Varah (Golub et al. 1999) show that the generalized eigenvalue problem

$$(H_1 - \lambda H_0) v = 0 \tag{5}$$

can be analyzed for errors to the first order. For a given eigenvalue λ and the associated eigenvector v, suppose

$$(H_1 + \epsilon \hat{H}_1)(v + \epsilon v^{(1)} + \cdots) = (\lambda + \epsilon \lambda^{(1)} + \cdots)(H_0 + \epsilon \hat{H}_0)(v + \epsilon v^{(1)} + \cdots)$$

is an ϵ-perturbation of our eigenvalue problem. Looking only at first order errors gives

$$(H_1 - \lambda H_0)v^{(1)} = (\lambda^{(1)} H_0 + \lambda \hat{H}_0 - \hat{H}_1)v. \tag{6}$$

Since v is both a left and right eigenvector (both H_0 and H_1 are symmetric), the left side of (6) is annihilated by multiplication on the left by v^{Tr}:

$$\lambda^{(1)} = \frac{v^{\mathrm{Tr}}(\hat{H}_1 - \lambda \hat{H}_0)v}{v^{\mathrm{Tr}} H_0 v}. \tag{7}$$

Assume the perturbations are of the same size as the precise value, that is, $\|\hat{H}_0\|_2 = \|H_0\|_2$ and $\|\hat{H}_1\|_2 = \|H_1\|_2$, and v is normalized as a unit vector, then (7) gives the error bound

$$\|\lambda^{(1)}\| \leq \frac{\|H_1\|_2 + \|\lambda\| \|H_0\|_2}{|v^{\mathrm{Tr}} H_0 v|}.$$

Assuming $\|H_1\|_2 = \|H_0\|_2$ (which is reasonable since the matrices have such close structures) and recalling λ is a root of unity so that $\|\lambda\| = 1$ give

$$\|\lambda^{(1)}\| \leq \frac{2\|H_0\|_2}{|v^{\mathrm{Tr}} H_0 v|}. \tag{8}$$

Since the norm of H_0 can always be bounded by scaling when necessary, the interesting quantity in the error bound (8) is

$$\frac{1}{|v^{\mathrm{Tr}} H_0 v|}. \tag{9}$$

The coefficients c_j's play a role in bounding the errors of the generalized eigenvalues. Notice that the columns of $(V^T)^{-1}$ give both the right and left eigenvectors of (5). If λ_j is the eigenvalue corresponding with the j-th column of $(V^T)^{-1}$, that is $v_j = (V^T)^{-1} e_j$ for (5), then (9) can be reduced to

$$\frac{|v_j^{\mathrm{Tr}} \cdot v_j|^2}{|v_j^{\mathrm{Tr}} H_0 v_j|} = \frac{\|v_j\|^2}{|v_j^{\mathrm{Tr}} \cdot V \cdot D \cdot V^{\mathrm{Tr}} \cdot v_j|} = \frac{\|(V^T)^{-1} e_j\|^2}{|c_j|} \leq \frac{\|(V^T)^{-1}\|^2}{|c_j|}.$$

An eigenvalue is *well-disposed* (Golub et al. 1999) if the quantity (9) is "small". From (8) we see that a well-disposed eigenvalue will result in a small error in the computation of the associated eigenvalue.

The advantage of solving the generalized eigenvalue problem in (4) is that there are stable numerical methods for such a problem and those methods do not require $b_1, \ldots, b_n$ to be on a unit circle (Golub et al. 1999).

3.3 Separation of powers

After determining the term values b_j's, we still needs to consider the precision required for correctly recovering the integer exponents through taking the logarithms of $b_j = \omega^{d_j}$ with $\omega = \exp(2\pi i/m)$.

Since two consecutive m-th roots of unity on the unit circle are separated by an angle of radian $\frac{2\pi}{m}$, the distance between these two points is bounded below by twice the sine of half the angle between them. Thus, in order to separate any two such points by rounding one must have values correct to

$$\frac{1}{2}|2\sin(\frac{\pi}{m})| \approx \frac{\pi}{m} \text{ and } m = p_1 \cdots p_n$$

with p_k primes and $p_k > \deg f_{x_k}$.

3.4 Recovering the coefficients

Once the term values b_j's have been determined, it still remains to compute their coefficients c_j's, which can be done in a number of ways (Golub et al. 1999). If the term values are determined as general eigenvalues in (5) by the QZ algorithm, the computed eigenvectors v_j can be put to use here. Let M be the matrix whose columns are eigenvectors v_j, then the coefficients can be computed by

$$c_j = (v_j^T H_0 v_j)(T_{j,1})^2,$$

with $T = M^{-1} = S^{-1}V^T$ and S a diagonal scaling matrix (Golub et al. 1999).

On the other hand, we can directly solve the Vandermonde system (2) to determine c_j's.

The conditioning of the associated Vandermonde system

While Vandermonde matrices tend to have poor conditioning, especially for real number data (Beckermann 2000; Gautschi and Inglese 1988), our problem can be much more behaved because all our points lie on the unit circle. For example, for a $t \times t$ Vandermonde matrix, if the nodes happen to be all the t-th roots of unity, the condition number for the 2-norm is 1, which is optimal (Gautschi 1975, Example 6.4).

Another kind of well behaved Vandermonde matrices on the unit circle has been studied in (Córdova et al. 1990). They consider a *Van der Corput sequence*, that is, $\{a_j\}_{j=0}^{\infty}$ for

$$a_j = \sum_{k=0}^{\infty} j_k 2^{-(k+1)} \text{ with } j = \sum_{k=0}^{\infty} j_k 2^k \text{ for } j_k \in \{0, 1\}.$$

For a $t \times t$ Vandermonde matrix with nodes $\exp(2\pi i a_0)$, ..., $\exp(2\pi i a_{t-1})$, the 2-norm condition number is less than $\sqrt{2t}$ (Córdova et al. 1990, Corollary 3).

In general, the conditioning of a Vandermonde matrix depends on the distance between different nodes, which follows from a well-known explicit formula for the inverse. If V is a $t \times t$ Vandermonde matrix with nodes $z_1, \ldots, z_t$, then $V^{-1} = [a_{j,k}]$ where

$$\ell_j(z) = a_{j,1} + a_{j,2}z + \cdots + a_{j,t}z^{t-1} = \prod_{\substack{u=1 \\ u \neq j}}^{t} \frac{z - z_u}{z_j - z_u}$$

is the j-th Lagrange polynomial of points $z_1, \ldots, z_t$. In our case, $z_1, \ldots, z_t$ are all on the unit circle. We have the following upper and lower bounds for $\|V^{-1}\|$ in the infinity norm (Gautschi 1975):

$$\max_{1 \le j \le t} \prod_{\substack{u=1 \\ u \neq j}}^{t} \frac{1}{|z_j - z_u|} < \|V^{-1}\|_\infty \le \max_{1 \le j \le t} \prod_{\substack{u=1 \\ u \neq j}}^{t} \frac{2}{|z_j - z_u|}.$$

Now we look at the condition of our associated Vandermonde system. Suppose $p_1, \ldots, p_n$ are distinct primes, $p_k > \deg_{x_k} f$, and $\omega = \exp(2\pi i/m)$ for $m = p_1 \cdots p_n$. If the target polynomial f is evaluated at powers of $(\omega_1, \ldots, \omega_n)$ for $\omega_k = \omega^{m/p_k}$, the distribution of term values on the unit circle is fixed because the polynomial terms are fixed.

To scrutinize the distribution of term values, instead of $f = \sum_{j=1}^{t} c_j x_1^{d_{j_1}} \cdots x_n^{d_{j_n}}$, we consider the univariate $f_1(x) = \sum_{j=1}^{t} c_j x^{d_j}$ with $d_j = d_{j_1} \cdot (m/p_1) + \cdots + d_{j_n} \cdot (m/p_n)$. Now the term values are $\omega^{d_1}, \ldots, \omega^{d_n}$, and their distribution on the unit circle depends on m and the differences between exponents d_j and d_k for $j \neq k$. If some of them are bunched together, then comparing to m some d_j's are relatively close to each other, which may lead to the ill-conditioning of the Vandermonde system.

Therefore, we propose to randomize the distribution of term values on the unit circle. Under the condition that $p_1, \ldots, p_n$ are given (hence their product m), we randomly pick an integer r such that $0 < r < \min(p_1, \ldots, p_n)$ and consider f evaluated at powers of $(\omega_1^r, \ldots, \omega_n^r)$. Now the corresponding univariate f_1 are evaluated at powers of ω^r, but it can also be viewed as another univariate polynomial $f_1^{[r]} = \sum_{j=1}^{t} c_j x^{d_j \cdot r}$ evaluated at powers of ω. In other word, the difference between ever pair of exponents is now multiplied by a random r in mod m, and may produce a distribution of term values that provides a better condition for the Vandermonde system. (As for the interpolation result, the original polynomial terms can be recovered by dividing each exponent by r in mod m.)

We implement the randomization strategy as the following: the user determines an integer threshold $\zeta > 0$ and randomly picks ζ distinct integers $r_1, \ldots, r_\zeta$ such that $0 < r_k < \min(p_1, \ldots, p_n)$ for each interpolation attempt. It is very likely that at least one of these ζ interpolation results is based on the computation of a "well-distributed" term values. If $\zeta \ge 2$, we may check whether there are two (or more) interpolation results that are very closed to each other. Such results are likely to be good approximations of the target polynomial.

We can further randomize the multiples of differences for exponents in each variable. For each interpolation attempt, we pick n random integers $r_1, \ldots, r_n$ such that $0 < r_k < p_k$ for $1 \leq k \leq n$, then interpolate the polynomial f evaluated at the powers of $(\omega_1^{r_1}, \ldots, \omega_n^{r_n})$. Rather than being multiplied by a random r in mod m, the exponent difference $d_j - d_k$ becomes a sum of random multiples of its components in mod m. That is,

$$r_1 \cdot (d_{j_1} - d_{k_1}) \cdot \left(\frac{m}{p_1}\right) + \cdots + r_n \cdot (d_{j_n} - d_{k_n}) \cdot \left(\frac{m}{p_n}\right).$$

Similarly, to recover the original terms in the interpolation result, the exponents in each variable x_k are divided by r_k accordingly.

4 Future directions

We have implemented our methods in Maple and are currently conducting experiments. A full sensitivity analysis, to obtain stronger guarantees, is being pursued. Furthermore, the generalized eigenvalue approach may be exploited for interpolation at real values.

On the other hand, based on the polynomial relations between trigonometric functions, we have extended our sparse interpolation methods to the interpolation of trigonometric functions. We intend to further investigate the sensitivity and experiments in this context as well.

Acknowledgements

We thank Annie Cuyt and Brigitte Verdonk for pointing recent related works. We thank Bernhard Beckermann for his comments.

References

Beckermann, B., 2000. The condition number of real Vandermonde, Krylov and positive definite Hankel matrices. Numeriche Mathematik 85, 553–577.

Ben-Or, M., Tiwari, P., 1988. A deterministic algorithm for sparse multivariate polynomial interpolation. In: Proc. Twentieth Annual ACM Symp. Theory Comput. ACM Press, New York, N.Y., pp. 301–309.

Brezinski, C., 1991. History of Continued Fractions and Padé Approximants. Springer Verlag, Heidelberg, Germany.

Córdova, A., Gautschi, W., Ruscheweyh, S., 1990. Vandermonde matrices on the circle: spectral properties and conditioning. Numerische Mathematik 57, 577–591.

Corless, R. M., Giesbrecht, M. W., Kotsireas, I. S., Watt, S. M., 2001. Numerical implicitization of parametric hypersurfaces with linear algebra. In: Roanes-Lozano, E. (Ed.), Artificial Intelligence and Symbolic Computation: International Conference AISC 2000. Springer Verlag, Heidelberg, Germany, pp. 174–183.

Díaz, A., Kaltofen, E., 1998. FoxBox a system for manipulating symbolic objects in black box representation. In: Gloor, O. (Ed.), Proc. 1998 Internat. Symp. Symbolic Algebraic Comput. (ISSAC'98). ACM Press, New York, N. Y., pp. 30–37.

Gautschi, W., 1975. Norm estimates for inverses of Vandermonde matrices. Numerische Mathematik 23, 337–347.

Gautschi, W., Inglese, G., 1988. Lower bounds for the condition numbers of Vandermonde matrices. Numerische Mathematik 52, 241–250.

Geddes, K. O., Czapor, S. R., Labahn, G., 1992. Algorithms for Computer Algebra. Kluwer Academic Publ., Boston, Massachusetts, USA.

Golub, G. H., Milanfar, P., Varah, J., 1999. A stable numerical method for inverting shape from moments. SIAM J. Sci. Comput. 21 (4), 1222–1243.

Grigoriev, D. Y., Karpinski, M., Singer, M. F., 1990. Fast parallel algorithms for sparse multivariate polynomial interpolation over finite fields. SIAM J. Comput. 19 (6), 1059–1063.

Hildebrand, F. B., 1956. Introduction to numerical analysis. McGraw-Hill Book Co., New York.

Kaltofen, E., Lakshman Yagati, 1988. Improved sparse multivariate polynomial interpolation algorithms. In: Gianni, P. (Ed.), Symbolic Algebraic Comput. Internat. Symp. ISSAC '88 Proc. Vol. 358 of Lect. Notes Comput. Sci. Springer Verlag, Heidelberg, Germany, pp. 467–474.

Kaltofen, E., Trager, B., 1990. Computing with polynomials given by black boxes for their evaluations: Greatest common divisors, factorization, separation of numerators and denominators. J. Symbolic Comput. 9 (3), 301–320.

Mansour, Y., 1995. Randomized approxmation and interpolation of sparse polynomials. SIAM Journal on Computing 24 (2), 357–368.

Marple, Jr., S. L., 1987. Digital Spectral Analysis. Prentice-Hall, Englewood Cliffs, N.J.

Milanfar, P., Verghese, G. C., Karl, W. C., Wilsky, A. S., 1995. Reconstructing polygons from moments with connections to array processing. IEEE Trans. Signal Processing 43 (2), 432–443.

Prony, R., Floréal et Prairial III (1795). Essai expérimental et analytique sur les lois de la Dilatabilité des fluides élastique et sur celles de la Force expansive de la vapeur de l'eau et de la vapeur de l'alkool, à différentes températures. J. de l'École Polytechnique 1, 24–76, R. Prony is Gaspard(-Clair-François-Marie) Riche, baron de Prony.

Wilkinson, J. H., 1963. Rounding errors in algebraic processes. Prentice-Hall, Englewood Cliffs, N.J.

Zilic, Z., Radecka, K., 1999. On feasible multivariate polynomial interpolations over arbitrary fields. In: Dooley, S. (Ed.), ISSAC 99 Proc. 1999 Internat. Symp. Symbolic Algebraic Comput. ACM Press, New York, N. Y., pp. 67–74.

Zippel, R., 1979. Probabilistic algorithms for sparse polynomials. In: Proc. EUROSAM '79. Vol. 72 of Lect. Notes Comput. Sci. Springer Verlag, Heidelberg, Germany, pp. 216–226.

Zippel, R., 1990. Interpolating polynomials from their values. J. Symbolic Comput. 9 (3), 375–403.