Numerical Algorithms manuscript No.
(will be inserted by the editor)

Polynomial Algebra for Birkhoff Interpolants

**John Butcher · Robert M. Corless ·
Laureano Gonzalez-Vega · Azar Shakoori**

Received: date / Accepted: date

Abstract We introduce a unifying formulation of a number of related problems which can all be solved using a contour integral formula. Each of these problems requires finding a non-trivial linear combination of some of the values of a function f, and its first and higher derivatives, at a number of data points. This linear combination is required to have zero value when f is a polynomial of up to a specific degree p. Examples of this type of problem include Lagrange, Hermite and Hermite-Birkhoff interpolation; and various numerical quadrature and differentiation formulae. Other applications include the estimation of missing data and root-finding.

Keywords Lagrange, Hermite, and Hermite-Birkhoff interpolation · Contour integrals · Barycentric form · Root-finding

Mathematics Subject Classification (2000) 41A05 · 65D05 · 65D25 · 65D30

1 Introduction

The purpose of this paper is to present a number of approximation formulae using a single standard formulation. These include formulae for Lagrange, Hermite and Hermite-Birkhoff interpolation (hereafter HB interpolation), divided difference formulae, numerical quadrature and numerical differentiation. Each of these approximation formulae can be written as a non-trivial linear combination

$$\sum_{i=1}^{n} \sum_{j \in S_i} \widehat{a}_{ij} f^{(j)}(\tau_i) = 0, \qquad (1)$$

where f is a polynomial of degree not exceeding p, and S_i is a subset of $\{0, 1, \ldots, s_i - 1\}$ such that $s_i - 1 \in S_i$. The positive integers s_i are sometimes referred to as 'confluencies'. The value of p is $-2 + \sum_{i=1}^{n} |S_i|$, where $|S_i|$ is the cardinality of S_i.

R. M. Corless
Department of Applied Mathematics University of Western Ontario
Tel.:
Fax:
E-mail: rcorless@uwo.ca

If $S_i = \{0, 1, \ldots, s_i - 1\}$, for each $i = 1, 2, \ldots, n$, we will rewrite (1) in the form

$$\sum_{i=1}^{n} \sum_{j=0}^{s_i-1} a_{ij} f^{(j)}(\tau_i) = 0. \tag{2}$$

The common feature of these formulations is their relation to contour integrals of the form

$$\frac{1}{2\pi i} \oint_C R(z) f(z) dz, \tag{3}$$

where R is a rational function related to the problem. By using the Cauchy integral formula we will find the approximation we require.

The paper is organized as follows: a collection of pointers to known literature on HB interpolation and related problems will be presented in Section 2. This is followed by Section 3, where the central result is given and by Section 4 where diverse applications are considered. Results and discussion on the existence and other properties of these approximations are presented in Section 5.

2 Selections from the literature

The literature relevant to this present paper is substantial, and we do not present a comprehensive survey. The references of the papers we cite should be consulted for further pointers.

HB interpolation problems occur in several applications. For example, the numerical solution of initial-value problems for ordinary differential equations often needs interpolants for event location, error control, or simply for graphical output. Derivative data is often more readily available than data on the value of the solution; see e.g. [14] and, more recently, [31]. Birkhoff's original paper [3] was concerned with "mechanical differentiation and quadrature", and these remain important. Other, newer, applications occur in, for instance, Computer-Aided Geometric Design, and in any application that has to deal with missing or unsampled data.

Several algorithms exist in the literature for solving HB interpolation problems. The simplest is presented in one line in [23, p. 12], but is not a practical algorithm for even moderate-sized problems, being at best of cost cubic in the size of the input. Existing more practical algorithms include [28] and [27], but we claim that the algorithm presented in this paper is more numerically stable and in some cases much faster, especially if the number of missing data points is small compared to the overall number of data points.

The method of this paper is also useful in the derivation of other formulae, including compact finite differences [21], which themselves find applications in several fields (see e.g. [33,34]).

The main technique of this paper is contour integration, which has a long and intimate connection with interpolation and with finite differences; see, e.g. [26, p. 11]. We will cite such references as we need in the course of the exposition. This technique was first used for HB interpolation in [6], and we develop it further here.

HB interpolation is also an old subject, dating to 1906 with Birkhoff's first paper, and comprehensively reviewed till 1983 in [23]. We do not attempt a comprehensive review of the literature since then. Indeed, our paper is concerned with an algorithm for solving specific HB interpolation problems, and not with finding conditions for unique solvability of such problems, which represents a significant focus of modern research in the area. HB interpolation problems have several interesting properties. The first of these is that HB interpolation problems do not always have unique solutions. The continuity properties of HB interpolation problems are also of interest [10]. This will be discussed further in Section 5.

In this paper we rely on a particular solution of the Hermite interpolation problem, which can be regarded as a special case of the HB interpolation problem. It is well-known, see e.g. [15], that Hermite interpolation (via divided differences for example) can be used as a fast method to solve confluent Vandermonde matrices. Some of the present authors maintain in another paper that methods for solving Hermite interpolation problems that avoid using Newton or monomial bases may be more numerically stable [9], and that in particular algorithms that treat each point equally, and do not depend on an imposed ordering, may result in better-conditioned problems. In particular, following [2] and [16] and [8], we use the *barycentric forms*: If the (Hermite) data $f^{(j)}(\tau_i)/j!$ is known for $1 \le i \le n$ and $0 \le j \le s_i - 1$ (as in the previous section), then the interpolating polynomial $f(t)$ may be represented (in a numerically stable fashion, if the confluencies s_i are not too large) as

$$f(t) = w(t) \sum_{i=1}^{n} \sum_{j=0}^{s_i-1} \sum_{k=0}^{j} \gamma_{ij} f^{(j)}(\tau_i)(t-\tau_i)^{k-j-1}/j! \tag{4}$$

or (the second form)

$$f(t) = \frac{\sum_{i=1}^{n} \sum_{j=0}^{s_i-1} \sum_{k=0}^{j} \gamma_{ij} f^{(j)}(\tau_i)(t-\tau_i)^{k-j-1}/j!}{\sum_{i=1}^{n} \sum_{j=0}^{s_i-1} \gamma_{ij}(t-\tau_i)^{-j-1}} \,. \tag{5}$$

Evaluating $f(t)$ or its derivatives exactly (to floating-point precision) at the nodes $t = \tau_i$ requires special treatment, but this turns out not to be difficult. The coefficients γ_{ij} appearing in those formulae were computed via divided differences in [29]. Unfortunately, the numerical stability of that computation depends on the ordering of the nodes τ_i; the method we present below appears to have superior stability, in compensation for being a constant factor slower than the algorithm of [29].

Finally, our paper makes heavy use of partial fractions: specifically, computation of partial fraction expansions in the case when the denominator is completely factored into (possibly complex) linear factors. See, e.g. [13], although the algorithms we use go back at least to Euler (1738). The algorithms appropriate to this case are simpler than when the factors of the denominator are not known (see e.g. [25]) or when computation over other fields than the complex numbers are needed (see e.g. [5, 11, 12, 30]).

It seems remarkable that the connection between partial fractions and Hermite interpolation, which we learned from [29] but was 'in the literature' much before that—even Turnbull [32] states that he believes the result is not new, possibly going back to Jacobi in 1841—is not better known. Recent papers apparently rediscovering the connection include [19] and [24], which last uses it and the interpolating definition of matrix functions [17] to evaluate the matrix exponential in an efficient and robust way. Asymptotically fast (though numerically unstable) algorithms for computing the partial fraction decomposition in $O(n \log n \log\log n)$ operations date to [7, 20], and the FFT is again used in [1]. We do not use asymptotically fast methods, because we believe they are numerically unstable. The algorithm we use is (loosely speaking) of quadratic cost, not quasilinear.

3 Solution by contour integrals

3.1 Statement of the problem

We are concerned with a function f and various approximations related to it. The approximations will have order p, in the sense that if f is replaced by a polynomial of degree p, then the approximation will be exact. The approximations will all have the form

$$\sum_{i=1}^{n}\sum_{j=0}^{s_i-1} a_{ij} f^{(j)}(\tau_i) = 0, \quad \deg(f) \le p. \tag{6}$$

Because of its relation to Hermite interpolation, we will refer to the construction of approximations satisfying (6) as the “Hermite case”. In Subsection 3.2 we will obtain formulae for the coefficients a_{ij}.

A more general problem, which we will refer to as the “Birkhoff case”, is to assume that there is missing data. In other words, we will assume that for each i, $f^{(j)}(\tau_i)$ is given only for a subset S_i of $\{0, 1, \ldots, s_i-1\}$. We will assume that such a subset always contains $s_i - 1$. As a matter of parsimony, this will enable us to avoid the situation that s_i could have been replaced by a lower integer.[1] With this more general case, (6) will be replaced by

$$\sum_{i=1}^{n}\sum_{j\in S_i} \widehat{a}_{ij} f^{(j)}(\tau_i) = 0, \quad \deg(f) \le p. \tag{7}$$

By counting the number of constraints, we would hope to be able to solve (6) with $p = -2 + \sum_{i=1}^{n} s_i$. In the Birkhoff case, again by counting constraints, we see that it is appropriate to aim for an order

$$p = -2 + \sum_{i=1}^{n} |S_i|. \tag{8}$$

Conditions under which (7) is guaranteed to have a solution will be discussed in Section 5.

3.2 Solution using contour integrals in the Hermite case

Given an approximation problem in the Hermite case, we define $w(z)$ by

$$w(z) = \prod_{i=1}^{n} (z - \tau_i)^{s_i}, \tag{9}$$

where the τ_i and s_i are as in (6). We have the following lemma.

Lemma 1 *Let f be a polynomial of degree not exceeding $p = -2 + \sum_{i=1}^{n} s_i$. Then*

$$\frac{1}{2\pi i}\oint_C \frac{f(z)}{w(z)} dz = 0, \tag{10}$$

where $w(z)$ is defined by (9) and the contour C is a circle with centre 0 and radius R greater than $\max_{i=1}^{n} |\tau_i|$.

[1] Our Maple and Matlab codes make no such restriction, though—they simply do more work than strictly necessary and give an answer equivalent to the most parsimonious approach, although containing extra factors in numerator and denominator that cancel out.

Proof The proof is a classic result in complex analysis and is an easy consequence of (for example) [22, Thm. 8.1, p. 233].

♮

We can now write $w(z)^{-1}$ in (10) in partial fractions

$$\frac{1}{w(z)} = \sum_{i=1}^{n} \sum_{j=0}^{s_i-1} \frac{\gamma_{ij}}{(z-\tau_i)^{j+1}}, \tag{11}$$

so that substituting (11) into (10) and evaluating term by term, we find the result

$$\sum_{i=1}^{n} \sum_{j=0}^{s_i-1} \frac{\gamma_{ij}}{j!} f^{(j)}(\tau_i) = 0. \tag{12}$$

Hence, we can solve (6) using

$$a_{ij} = \frac{\gamma_{ij}}{j!}. \tag{13}$$

The γ_{ij} in (13) are known as the generalized barycentric weights in the Hermite interpolation problem [29].

To evaluate the a_{ij} in a convenient form, we use the following result, which is equivalent to the local series method for computing the partial fraction expansion of $1/w(z)$. Many sources contain this algorithm, for example [13, v. 1, p. 555]. Faster algorithms are known, see e.g. [29], but they can be significantly less stable numerically. We include a description of the algorithm here, for convenience; our MATLAB and MAPLE implementations of this algorithm use the recurrence relation (15) below.

Lemma 2 *Let K denote a finite set of integers, and define $\chi_1, \chi_2, \ldots$ by the formula*

$$\chi_0 + \chi_1 t + \chi_2 t^2 + \ldots = \chi_0 \prod_{k\in K} (1 - t\theta_k)^{-s_k}, \tag{14}$$

where χ_0 is given. Then, $\chi_1, \chi_2, \ldots$ satisfy the recurrence relation

$$\chi_j = \frac{1}{j} \sum_{\ell=0}^{j-1} \chi_\ell \phi_{j-\ell}, \tag{15}$$

where $\phi_j = \sum_{k\in K} s_k \theta_k^j$.

Proof From the logarithmic series we have

$$\ln(1 - \theta_k t) = -\sum_{j=1}^{\infty} \frac{t^j}{j} \theta_k^j,. \tag{16}$$

For $|t|$ sufficiently small, it follows by multiplying by $-s_k$ and summing over $k \in K$, that

$$\ln\Big(\prod_{k\in K} (1 - t\theta_k)^{-s_k}\Big) = \sum_{j=1}^{\infty} \frac{t^j}{j} \phi_j, \tag{17}$$

and therefore

$$\chi_0 + \chi_1 t + \chi_2 t^2 + \cdots = \chi_0 \exp\left(\sum_{j=1}^{\infty} \frac{t^j}{j}\phi_j\right). \tag{18}$$

Differentiate (18) with respect to t and it follows that

$$\chi_1 t + 2\chi_2 t + 3\chi_3 t^2 + \cdots = (\phi_1 + \phi_2 t + \cdots)(\chi_0 + \chi_1 t + \chi_2 t^2 + \cdots). \tag{19}$$

Equate the coefficients of t^{j-1} and (15) follows.

♮

In our application of this lemma, let

$$K = \{1, 2, \ldots, n\} \setminus \{i\}, \quad \theta_k = \frac{1}{\tau_k - \tau_i}, \quad \chi_0 = \prod_{k\in K} (\tau_i - \tau_k)^{-s_k}, \quad t = z - \tau_i.$$

For a given $i \in \{1, 2, \ldots, n\}$, evaluate the Laurent expansion about $z = \tau_i$ as follows

$$\begin{aligned} \frac{1}{w(z)} &= \frac{1}{(z-\tau_i)^{s_i}} \prod_{k\in K} (z-\tau_k)^{-s_k} \\ &= \frac{1}{(z-\tau_i)^{s_i}} \prod_{k\in K} \Big((\tau_i - \tau_k)\Big(1 - \frac{z-\tau_i}{\tau_k - \tau_i}\Big)\Big)^{-s_k} \\ &= \frac{1}{(z-\tau_i)^{s_i}} \chi_0 \prod_{k\in K} (1 - t\theta_k)^{-s_k} \\ &= \frac{1}{(z-\tau_i)^{s_i}} (\chi_0 + \chi_1(z-\tau_i) + \chi_2(z-\tau_i)^2 + \cdots) \end{aligned} \tag{20}$$

and hence

$$\gamma_{ij} = \chi_{s_i - j - 1}, \quad a_{ij} = \frac{\chi_{s_i-j-1}}{j!}. \tag{21}$$

Having found the coefficients in (6), we generalize to the Birkhoff case.

3.3 Solution in the Birkhoff case

To allow for missing data, replace (10) by

$$\frac{1}{2\pi i} \oint_C \frac{B(z) f(z)}{w(z)} dz = 0, \tag{22}$$

where

$$B(z) = b_0 + b_1 z + \cdots + b_m z^m \tag{23}$$

and m is the number of missing data items. The integral is still zero if f is now restricted to a polynomial of degree

$$p = -2 - m + \sum_{i=1}^{n} s_i = -2 + \sum_{i=1}^{n} |S_i|. \tag{24}$$

The coefficients in $B(z)$, suitably normalized, are to be chosen so that, in the partial fraction expansion of $B(z)/w(z)$, which we will write as

$$\frac{B(z)}{w(z)} = \sum_{i=1}^{n} \sum_{j=0}^{s_i-1} \frac{\widehat{\gamma}_{ij}}{(z-\tau_i)^{j+1}}, \tag{25}$$

the value of $\widehat{\gamma}_{ij}$ will vanish whenever $j \notin S_i$. To solve (7), it is then possible to use a slightly modified form of Lemma 1 to give the result

$$\widehat{a}_{ij} = \frac{\widehat{\gamma}_{ij}}{j!}, \quad j \in S_i. \tag{26}$$

We compute the coefficients $\widehat{\gamma}_{ij}$ by multiplying the local series for $B(z)$ with the partial fraction decomposition of $1/w(z)$. The coefficients are computed with Cauchy convolution, which we write in matrix notation in the Theorem below.

Theorem 1 *For $i = 1, 2, \ldots, n$, let $\mathbf{M}_i$ denote the $s_i \times (m+1)$ matrix*

$$\mathbf{M}_i = \mathbf{\Gamma}_i \mathbf{T}_i, \tag{27}$$

where $\mathbf{\Gamma}_i$ is the $s_i \times s_i$ matrix

$$\mathbf{\Gamma}_i = \begin{bmatrix} \gamma_{i,0} & \gamma_{i,1} & \cdots & \gamma_{i,s_i-2} & \gamma_{i,s_i-1} \\ \gamma_{i,1} & \gamma_{i,2} & \cdots & \gamma_{i,s_i-1} & 0 \\ \vdots & & & & \vdots \\ \gamma_{i,s_i-1} & 0 & \ldots & 0 & 0 \end{bmatrix}, \tag{28}$$

where for $0 \leqslant j \leqslant s_i - 1$ and $0 \leqslant k \leqslant s_i - 1$, the (j,k) entry (indexing from 0) of $\mathbf{\Gamma}_i$ is

$$\mathbf{\Gamma}_i(j,k) = \begin{cases} \gamma_{i,k+j} & \text{if } k + j \leqslant s_i - 1 \\ 0 & \text{otherwise} \end{cases} \tag{29}$$

and $\mathbf{T}_i$ is the $s_i \times (m+1)$ matrix

$$\mathbf{T}_i = \begin{bmatrix} 1 & \tau_i & \tau_i^2 & \ldots & \tau_i^m \\ & 1 & 2\tau_i & \ldots & m\tau_i^{m-1} \\ & & 1 & \ldots & \frac{m(m-1)}{2!}\tau_i^{m-2} \\ & & & \ddots & \vdots \end{bmatrix}, \tag{30}$$

with the (j,k) element

$$\mathbf{T}_i(j,k) = \binom{k}{j} t_i^{k-j}, \tag{31}$$

and the last row allows computation of the (s_i-1)th derivative of B evaluated at τ_i. [Recall that $\binom{k}{j} = 0$, if $j > k$.] Let $\widetilde{\mathbf{M}}$ denote the $m \times (m+1)$ matrix

$$\widetilde{\mathbf{M}} = \begin{bmatrix} \widetilde{M}_1 \\ \widetilde{M}_2 \\ \vdots \\ \widetilde{M}_n \end{bmatrix}, \tag{32}$$

where $\widetilde{M}_i$ consists of rows of $\mathbf{M}_i$ such that row number j is included if and only if $j \notin S_i$.

Then, if $\widetilde{\mathbf{M}}$ has rank m, the coefficients in (23) are a non-zero solution of

$$\widetilde{\mathbf{M}}\mathbf{b} = 0\,, \quad \textit{where} \quad \mathbf{b} = \left[\, b_0\; b_1\; \ldots\; b_m \,\right]^T. \tag{33}$$

Proof The proof requires finding m homogeneous linear equations in the elements of $\mathbf{b}$. This, in turn, requires finding the values of $\widetilde{\gamma}_{ij}$ in (26), in terms of $\mathbf{b}$. The first step is to write the Taylor coefficients about τ_i in terms of $\mathbf{b}$. This is given by the formula

$$\begin{bmatrix} B(\tau_i) \\ B'(\tau_i) \\ \frac{1}{2!}B''(\tau_i) \\ \vdots \\ \frac{1}{(s_i-1)!}B^{(s_i-1)}(\tau_i) \end{bmatrix} = \mathbf{T}_i\mathbf{b}. \tag{34}$$

The values of $\widehat{\gamma}_{ij}$, for $j = 0, 1, \ldots, s_i - 1$, are found by multiplying (11) by

$$B(z) = B(\tau_i) + B'(\tau_i)(z-\tau_i) + \cdots + \frac{1}{(s_i-1)!}B^{(s_i-1)}(\tau_i)(z-\tau_i)^{s_i-1} + O((z-\tau_i)^{s_i}). \tag{35}$$

[Note that some of those derivatives will be zero if $s_i > m$.] This gives

$$\widehat{\gamma}_{ij} = \sum_{\ell=0}^{\min(j,m)} \gamma_{i\ell}\frac{B^{(\ell)}(\tau_i)}{\ell!}, \tag{36}$$

which can be written as

$$\begin{bmatrix} \widehat{\gamma}_{i0} \\ \widehat{\gamma}_{i1} \\ \vdots \\ \widehat{\gamma}_{i,s_i-1} \end{bmatrix} = \mathbf{\Gamma}_i \begin{bmatrix} B(\tau_i) \\ B'(\tau_i) \\ \frac{1}{2!}B''(\tau_i) \\ \vdots \\ \frac{1}{(s_i-1)!}B^{(s_i-1)}(\tau_i) \end{bmatrix}. \tag{37}$$

Substitute from (34) and use (27); we find that

$$\begin{bmatrix} \widehat{\gamma}_{i0} \\ \widehat{\gamma}_{i1} \\ \vdots \\ \widehat{\gamma}_{i,s_i-1} \end{bmatrix} = \mathbf{\Gamma}_i(\mathbf{T}_i\mathbf{b}) = \mathbf{M}_i\mathbf{b} \tag{38}$$

and because $\widehat{\gamma}_{ij} = 0$ for $j \notin S_i$, it follows that $\widetilde{\mathbf{M}_i}\mathbf{b} = 0$. Combine this result for all $i = 1, 2, \ldots, n$ and the theorem follows.

♮

As in Subsection 3.2. we conclude this investigation of the Birkhoff case by finding the coefficients in(7). These are

$$\widehat{a}_{ij} = \frac{\widehat{\gamma}_{ij}}{j!}, \qquad i \in \{1, 2, \ldots, n\}, j \in S_i. \tag{39}$$

4 Specific applications

4.1 Interpolation

Given data points as in Section 3, we can obtain a formula for the interpolating polynomial by adding to the set $\{\tau_1, \tau_2, \ldots, \tau_n\}$ the additional point $\tau_0 = t$, with $s_0 = 1$. This means, in the Hermite case, that we make use of the contour integral (10) in which $w(z)$ is replaced by

$$w(z) = \prod_{i=0}^{n} (z - \tau_i)^{s_i} = (z - t) \prod_{i=1}^{n} (z - \tau_i)^{s_i} \tag{40}$$

Following through with the construction of Subsection 3.2, we find the modified form of (6) as follows:

$$a_{00}(t) f(t) + \sum_{i=1}^{n} \sum_{j=0}^{s_i - 1} a_{ij}(t) f^{(j)}(\tau_i) = 0, \quad \deg(f) \le -1 + \sum_{i=1}^{n} s_i. \tag{41}$$

Assuming that $a_{00}(t) \ne 0$, this gives the interpolation formula

$$f(t) = \sum_{i=1}^{n} \sum_{j=0}^{s_i - 1} \frac{-a_{ij}(t)}{a_{00}(t)} f^{(j)}(\tau_i), \quad \deg(f) \le -1 + \sum_{i=1}^{n} s_i. \tag{42}$$

A priori it is not clear that the expression for $f(t)$ is polynomial in t; we shall discuss later just why it is indeed polynomial. In the Birkhoff case, we can do the construction described in Subsection 3.3, again with $w(z)$ replaced by (40). This gives a formula

$$f(t) = \sum_{i=1}^{n} \sum_{j \in S_i} \frac{-\widehat{a}_{ij}(t)}{\widehat{a}_{00}(t)} f^{(j)}(\tau_i), \quad \deg(f) \le -1 + \sum_{i=1}^{n} |S_i|, \tag{43}$$

again assuming that $\widehat{a}_{00}(t) \ne 0$. Again this is polynomial in t.

If we have a particular numerical value of t, or only a few such values, for which we need to evaluate $f(t)$, then perhaps it makes sense to construct these coefficients $\widehat{a}_{00}(t)$ and $\widehat{a}_{ij}(t)$ anew for each t. If, however, we have a great many values of t for which we need to evaluate f, then perhaps it makes sense to do the construction "once and for all". This is indeed possible. The linear system defining the unknown coefficients of $B(z)$ then contains a matrix and right-hand-side that depend polynomially on the symbol t. Solving this system is then a *symbolic computation*, and therefore more expensive, but still feasible in many applications. An alternative approach that may be cheaper, filling in missing data, is discussed in Section 4.2.

Example 1 Let $\tau = [0, r, 1]$ with data given as $f(0)$, $f'(r)$, and $f(1)$. We do not yet specify r, but for the moment imagine it is a real number between 0 and 1. Finding a polynomial of degree 2 that fits this data is a HB problem. We begin by defining $w(z) = z(z-r)^2(z-1)$, adding a new node $\tau_0 = t$ and making all necessary changes in our notation to make this example clear, and computing the partial fraction decomposition of

$$\begin{aligned}\frac{1}{(z-t)w(z)} &= \frac{1}{r\,(r-1)\,(r-t)\,(z-r)^2} - \frac{1}{(r-1)^2\,(t-1)\,(z-1)} \\ &+ \frac{1}{t\,(r-t)^2\,(t-1)\,(z-t)} + \frac{1}{r^2tz} \\ &+ \frac{2\,r-t+2\,rt-3\,r^2}{(z-r)\,r^2\,(r-1)^2\,(r-t)^2}\,.\end{aligned} \tag{44}$$

The matrices of Theorem 1 that we need are, therefore, simple: $\widetilde{M}_0$ is empty, as is $\widetilde{M}_1$ and $\widetilde{M}_3$, but we must have one row present in $\widetilde{M}_2$, namely

$$\widehat{\gamma}_{2,0}(b_0(t)+b_1(t)r)+\widehat{\gamma}_{2,1}b_1(t)=0\,. \tag{45}$$

Enforcing this will ensure that the residue multiplying the unknown value of f at r will be zero. Clearly, we have one constraint and two unknowns b_0 and b_1, each of which depend on t; it is convenient to add a normalization condition, namely that the residue at $z=t$, our new symbolic node, is -1. This will allow easy isolation of the value of $f(t)$ from the resulting zero sum of residues. This residue is

$$\frac{b_0(t)+b_1(t)t}{w(t)}=-1\,. \tag{46}$$

The determinant of the 2×2 matrix of these equations (45,46) is

$$\mathrm{Det}=\frac{2r-1}{r^2(1-r)^2}\,, \tag{47}$$

which clearly identifies the problematic cases $r=0$ and $r=1$ when our formulation is structurally discontinuous, and the case $r=1/2$ when the problem is not regular (see Section 5).

If r is not 0, 1/2, or 1, the system can be solved for the unknowns and the resulting residues computed, giving the barycentric formula

$$f(t)=w\,(t)\left(\frac{(t+1-2\,r)\,f\,(0)}{t\,(2\,r-1)\,(t-r)^2}+\frac{f'\,(r)}{(2\,r-1)\,(t-r)^2}+\frac{(2\,r-t)\,f\,(1)}{(t-r)^2\,(2\,r-1)\,(t-1)}\right) \tag{48}$$

Notice that $w(t)=t(t-r)^2(t-1)$ contains a factor that exactly cancels each of the t, $t-1$, and $(t-r)^2$ factors in each term of the barycentric formula. The end result is a polynomial in t. One could, if one wanted, rewrite the polynomial and express it in the monomial basis. We stress that doing so is, in general, a bad idea.

Remark 1 When we are missing m pieces of data, we need only solve an m by m linear system, symbolic in that it contains the symbol t; here we solved an $m+1$ by $m+1$ system (symbolic both in t and in the parameter r) because we chose to normalize by making the residue at $z=t$ to be -1, instead of taking $B(z)$ to be monic, $B(z)=b_0(t)+z$. This is a matter of taste, which makes no practical difference in the method.

4.2 Filling in missing data

We discuss the problem of inserting some or all of the data which was not originally available in an instance of the Birkhoff case. The inserted data is to be calculated from the interpolating polynomial defined by the original HB data, which we assume is regular. This problem can be looked at in terms of the zero-valued linear combination of the original data and a single additional piece of data, which is to hold for all polynomials of up to degree $-1 + \sum_{i=1}^{n} |S_i|$. We illustrate the process by an example, but note that the algorithm has been implemented in a general MATLAB program, written by Piers Lawrence at our request, and in a MAPLE program written by RMC.

Example 2 Suppose that $\tau = [1, 2, 4]$, (this is equivalent to the previous example, with a linear change of variable and specifying $r = 1/3$) and that we know $f(1)$, $f'(2)$, and $f(4)$. We will use the residue method to find $f(2)$. Here, $w(z) = (z-1)(z-2)^2(z-4)$. Unlike the case where we wish to find f at an arbitrary point $z = t$, where we needed a degree $m = 1$ multiplier $B(z)$, here it suffices to take $B(z) = 1$. In general, we may use a degree $m - 1$ multiplier. To continue, $w(z)^{-1}$ has the partial fraction expansion

$$\frac{1}{(z-1)(z-2)^2(z-4)} = -\frac{1/3}{z-1} + \frac{1/4}{z-2} - \frac{1/2}{(z-2)^2} + \frac{1/12}{z-4}\,. \tag{49}$$

For f a polynomial of degree not exceeding 2 we now have

$$0 = \frac{1}{2\pi i}\oint_C \frac{f(z)dz}{w(z)} = -\frac{1}{3}f(1) + \frac{1}{4}f(2) - \frac{1}{2}f'(2) + \frac{1}{12}f(4) \tag{50}$$

which immediately gives

$$f(2) = \tfrac{4}{3}f(1) + 2f'(2) - \tfrac{1}{3}f(4)\,. \tag{51}$$

Of course, this particular problem could have been solved by other approaches as well, for example by divided differences [26].

The general approach is to add a single pair (I, J) to the set of known data, at a time. That is, we suppose we wish to identify $f^{(J)}(\tau_I)$, which is not given by any (i, j) in $1 \le i \le n$, $j \in S_i$ (but τ_I is one of the interpolating nodes, and so we cannot use the approach of the previous section, which required that $t \ne \tau_i$ for each i). Now carry out the construction in Subsection 3.3, but here we do not set to zero the row corresponding to the pair (I, J); this gives us only the remaining $m - 1$ conditions to satisfy. Thus, we need a multiplier $B(z)$ of only degree $m - 1$. For convenience, as in example 1, we may use a degree m multiplier and set the residue for (I, J) to be -1, if this is possible. The resulting sum of residues is (in general)

$$a_{IJ} f^{(J)}(\tau_I) + \sum_{i=1}^{n} \sum_{j \in S_i} a_{ij} f^{(j)}(\tau_i) = 0 \tag{52}$$

If $a_{IJ} \ne 0$, this will allow us to identify the (formerly) missing piece of data.

Remark 2 If we wish to identify *all* missing data, then we need to do this process m times, once for each missing data point. It is evident that the equations are the same for each time, with the only difference being that the residues are, one at a time, being set to -1. Thus the matrix (call it $\mathbf{B}$) is identical each time; only the right-hand side changes, being a column of all zeros except one -1. That is, the coefficients of $B(z)$ for each missing point are given by the columns of $-\mathbf{B}^{-1}$. Our implementations use this method, and always fill in all missing data.

Table 1 Relative error in a degree 23 HB problem, solved in MATLAB by the fill-in method.

```
tau  = cos( (0:n)*pi/n ); % nodes
t    = linspace( -1, 1, 20*n+1 );
f = @(x) sin(pi*x);
df = @(x) pi*cos(pi*x);
ddf = @(x) -pi^2*sin(pi*x);
rho  = [f(tau); df(tau); ddf(tau)]; % data
rho(1,6)=NaN;
rho(1,7) = NaN;
rho(2,8)=NaN;
r    = rho(:);
r = birkhoff_interp1(r,tau,3)';
[w,D]   = genbarywts( tau, 3 ); % get barycentric weights
[y,yp]  = hermiteval( r, t, tau, 3, w, D ); % evaluate interpolant
```

The connection between the interpolating polynomial and the fill-in result given by (52) is as follows.

Theorem 2 *If* a_{IJ} *in* (52) *is not zero, and* $\varphi(t)$ *is the unique interpolating polynomial determined by the (assumed regular) HB problem* $f^{(j)}(\tau_i)$, $j \in S_i$, $i \in \{1, 2, \ldots, n\}$, *then*

$$f^{(J)}(\tau_I) = \varphi^{(J)}(\tau_I). \tag{53}$$

Proof Since $\varphi(t)$ is a polynomial of degree not exceeding $-1 + \sum_{i=1}^{n} |S_i|$, (52) holds with f replaced by φ. That is,

$$a_{IJ}\varphi^{(J)}(\tau_I) + \sum_{i=1}^{n} \sum_{j \in S_i} a_{ij}\varphi^{(j)}(\tau_i) = 0. \tag{54}$$

Because of the interpolating property of φ, $\varphi^{(j)}(\tau_i) = f^{(j)}(\tau_i)$ for all $j \in S_i$, $i \in \{1, 2, \ldots, n\}$. Hence,

$$a_{IJ}\varphi^{(J)}(\tau_I) + \sum_{i=1}^{n} \sum_{j \in S_i} a_{ij} f^{(j)}(\tau_i) = 0. \tag{55}$$

Subtract (55) from (52), and the result follows.

♮

Example 3 In Table 1 we see some MATLAB statements defining a degree 23 ($27 - 1$ minus $m = 3$ missing data) HB interpolation problem, together with calls to MATLAB routines that we have written to solve such problems. In Figure 1 we see a plot of the relative error of the polynomial *evaluated as a barycentric Hermite polynomial using the recovered (filled-in) data*. When we solved this problem using a symbolic evaluation point t using only 14 decimal digits for the computation of the coefficients and the solution of the mixed symbolic-numeric linear system, we found that the final formula contained significant numerical errors. In particular, there was a spurious pole in the barycentric formula. Increasing the number of digits to 20 cured the problem in this instance, but indicates in general that one may expect the 'fill-in' numerical method

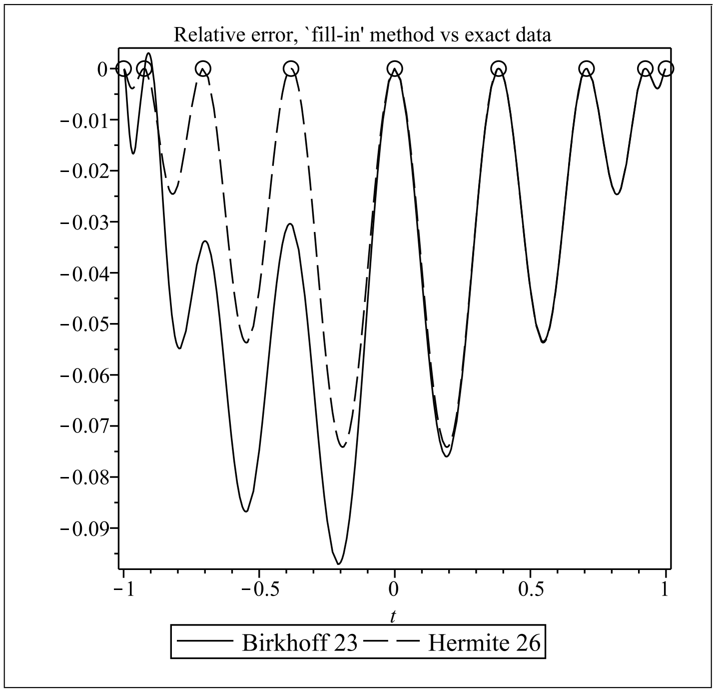

Fig. 1 Relative error $y/\sin(\pi t) - 1$ in a degree 23 HB interpolant (missing two function values and one derivative value). For comparison, we plot on the same figure the relative error in the corresponding degree 26 Hermite interpolant. One sees that the Hermite interpolant matches all 9 data points and their first and (less visibly) their second derivatives; one also sees in the graph which two function values were dropped for this example, and, looking carefully, one can also see that which derivative value was dropped. The change induced by the missing data is noticeable but not alarming.

to have greater numerical stability than the 'construct the polynomial' approach. This perhaps counter-intuitive observation will be discussed in greater detail later.

When the number of missing data is small, it is convenient as well as efficient to proceed in this manner. The MATLAB code `birkhoff_interp1` is available on request, and the other codes are available at `http:// www.orcca.on.ca/ TechReports/ 2007/ TR-07-05.html`.

4.3 Numerical quadrature and differentiation

4.4 Root-finding

We may wish to compute the roots of a univariate polynomial that is specified by a HB interpolation problem. There are strong numerical and structural advantages in working with the given data directly without completing the interpolation and converting to the familiar monomial basis [9]. Numerically stable sets of basis-preserving root-finding algorithms, in the Hermite and Lagrange cases, have been used in [8].

In a similar fashion, if we have a HB interpolation problem, we do not wish to first compute a monomial basis representation for the solution before finding the desired roots. Instead, we suggest that the approach of Section 4.2 be used to first fill in the missing data. Having done this, without completing the interpolation, we have converted the original problem to the Hermite case. We can then use existing algorithms to compute all roots of the given polynomial by solving a generalized eigenvalue problem. This is done by constructing a pair of regular matrices, known as the companion pencil matrices. The finite generalized eigenvalues of this matrix pencil are the roots of the original polynomial. We demonstrate this process with an example.

Example 4 As in example 2, let $\tau = [1, 2, 4]$. Suppose also that $f(1) = 1$, $f'(2) = 0$, and $f(4) = -1$. The 'fill-in' process gives

$$f(2) = \tfrac{4}{3} f(1) + 2f'(2) - \tfrac{1}{3} f(4) = \tfrac{5}{3}\,. \tag{56}$$

The generalized companion matrix pencil of [8] is

$$C_0 = \begin{bmatrix} 1 & & & & 1 \\ & 2 & & & \frac{5}{3} \\ & 1 & 2 & & 0 \\ & & & 4 & -1 \\ \frac{1}{3} & -\frac{1}{4} & \frac{1}{2} & -\frac{1}{12} & 0 \end{bmatrix} \tag{57}$$

and C_1 is, as usual, the 5×5 identity matrix with the $(5, 5)$ corner being replaced by zero. Note that the generalized barycentric weights γ_{ij} appear in the bottom row, and the polynomial values appear in the final column. The nodes appear in transposed Jordan blocks along the diagonal. It is easy to see that $\det(tC_1 - C0) = f(t)$ for any t. We find the generalized eigenvalues numerically. There are three infinite eigenvalues (because f is degree 2 while the matrices are 5 by 5), and two finite eigenvalues: Maple's call to LAPACK routines gets 3.58113883008418021 and 0.418861169915851983. [The exact values are $2 \pm \sqrt{10}/2$.]

We remark that the roots were found *without* converting to a monomial basis at any time.

4.5 Rational interpolation

The approach described in this paper can be generalized to the rational polynomial case, with fixed denominator. This is interesting and useful because, as in [8], one can choose the denominator with parameters that can be *tuned* to ensure monotonicity, convexity, or other properties, as in [4]. In what follows, we assume $R(z) = f(z)/q(z)$ is the rational polynomial with $q(z)$ fixed and not zero at any node. For compactness, we use superscripts for derivatives instead of the clumsier Leibniz notation:

$$R^{(\ell)}(\tau_i) = \frac{d^\ell}{dz^\ell} \left(\frac{f(z)}{q(z)} \right) \Big|_{\{z=\tau_i\}}\,. \tag{58}$$

Lemma 3 *Absorbing a denominator into the barycentric weights. Let* $w(z)$ *be defined as in* (9)*. Suppose* $\deg q(z) \leqslant -1 + \sum_{i=1}^{n} s_i$*. Put* $\sigma_{ij} = \frac{q^{(j)}(\tau_i)}{j!}$ *for brevity. Then*

$$\frac{q(z)}{w(z)} = \sum_{i=1}^{n} \sum_{j=0}^{s_i-1} \frac{\alpha_{ij}}{(z-\tau_i)^{j+1}} \tag{59}$$

where

$$\alpha_{ij} = \sum_{k=0}^{s_i-1-j} \gamma_{i,j+k}\sigma_{ik}\,. \tag{60}$$

Moreover, if $q(\tau_i) \neq 0$*, and* γ_{i,s_i-1} *is as defined in* (21) *then*

$$\alpha_{i,s_i-1} = \gamma_{i,s_i-1}\sigma_{i,0} \neq 0. \tag{61}$$

Proof By the Hermite interpolation formula,

$$\frac{q(z)}{w(z)} = \sum_{i=1}^{n} \sum_{j=0}^{s_i-1} \sum_{k=0}^{j} \gamma_{ij}\sigma_{ik}(z-\tau_i)^{k-j-1} \tag{62}$$

Interchanging the order of the second and third sums,

$$\frac{q(z)}{w(z)} = \sum_{i=1}^{n} \sum_{k=0}^{s_i-1} \sum_{j=k}^{s_i-1} \gamma_{ij}\sigma_{ik}(z-\tau_i)^{k-j-1}\,. \tag{63}$$

Putting $\ell = j - k$,

$$\frac{q(z)}{w(z)} = \sum_{i=1}^{n} \sum_{k=0}^{s_i-1} \sum_{\ell=0}^{s_i-1-k} \gamma_{i,\ell+k}\sigma_{ik}(z-\tau_i)^{-\ell-1} \tag{64}$$

Interchanging the second and third sums again and renaming the temporary variable ℓ as j again, we have

$$\frac{q(z)}{w(z)} = \sum_{i=1}^{n} \sum_{j=0}^{s_i-1} \left(\sum_{k=0}^{s_i-1-j} \gamma_{i,j+k}\sigma_{ik} \right) (z-\tau_i)^{-j-1} \tag{65}$$

as claimed. Now,

$$\alpha_{i,s_i-1} = \sum_{k=0}^{0} \gamma_{i,s_i-1+k}\sigma_{i,k} = \gamma_{i,s_i-1}\sigma_{i,0} \neq 0\,, \tag{66}$$

also as claimed, since $\sigma_{i,0} = q(\tau_i) \neq 0$.

♮

Lemma 4 *If* $\deg q(z) \leqslant -2 - m + \sum_{i=1}^{n} s_i$*, and if the contour* C *is taken large enough to enclose all* τ_i*, then for all polynomials* $B(z)$ *of degree* $\leqslant m$ *and all polynomials* $f(z)$ *of degree* $\leqslant -2 - m + \sum_{i=1}^{n} s_i$*, we have*

$$0 = \frac{1}{2\pi i} \oint_C \frac{B(z)f(z)}{w(z)} dz = \frac{1}{2\pi i} \oint_C \frac{B(z)q(z)}{w(z)} \cdot \frac{f(z)}{q(z)} dz\,. \tag{67}$$

Proof The proof follows by degree counting and Lemma 1.

♮

Remark 3 The degree restriction on $q(z)$ above is unnecessary. One can have arbitrarily large degrees for q, and the residues at infinity all turn out to be zero, leaving the same expansion. One can take $q(z)$ modulo $w(z)$ without harm.

Theorem 2 can be generalized to the rational polynomial case.

Theorem 3 *Let, as before, $R(z) = f(z)/q(z)$ be the rational polynomial with $q(z)$ fixed. Fix (I, J) as in Theorem 2, thus focussing our attention on a single missing data piece. Use superscripts to indicate that $B(z) = B^{[IJ]}(z)$ depends on the particular index pair (I, J). If $B(z)$ can be chosen as*

$$B(z) = B^{[IJ]}(z) = \sum_{k=0}^{m-1} b_k^{[IJ]} z^k \tag{68}$$

with

$$\frac{B^{[IJ]}(z)q(z)}{w(z)} = \frac{-1}{(z-\tau_I)^{J+1}} + \sum_{i=1}^{n} \sum_{j \in S_i} \frac{\alpha_{ij}^{[IJ]}}{(z-\tau_i)^{j+1}}\,, \tag{69}$$

where $J \notin S_I$, and as before m is the number of missing data items, then

$$\frac{R^{(J)}(\tau_i)}{J!} = \sum_{i=1}^{n} \sum_{j \in S_i} \alpha_{ij}^{[IJ]} \rho_{ij} \tag{70}$$

for every polynomial $f(z)$ of degree $\leqslant -1 - m + \sum_{i=1}^{n} S_i$ and with

$$\frac{R^{(j)}(\tau_i)}{j!} = \rho_{ij} \tag{71}$$

given for $1 \leqslant i \leqslant n$, $j \in S_i$.

Proof Replace m with $m-1$ in Lemma 4 (note that the residues when $q(z) = 0$ are zero), and apply the Cauchy Residue Theorem. It follows that

$$\begin{aligned} 0 =& \frac{1}{2\pi i} \oint \frac{B(z)q(z)}{w(z)} \cdot \frac{f(z)}{q(z)} dz \\ =& \frac{1}{2\pi i} \oint \left(\frac{-1}{(z-\tau_I)^{(J+1)}} + \sum_{i=1}^{n} \sum_{j \in S_i} \frac{\alpha_{ij}^{[IJ]}}{(z-\tau_i)^{j+1}} \right) \frac{f(z)}{q(z)} dz \\ =& \frac{-R^{(J)}(\tau_i)}{J!} + \sum_{i=1}^{n} \sum_{j \in S_i} \alpha_{ij}^{[IJ]} \rho_{ij}\,. \end{aligned} \tag{72}$$

♮

Remark 4 As in Section 4.2, one may find all missing data at once by inverting a single matrix.

5 Regularity and conditioning

The HB interpolation problem is not always *regular*, by which is commonly meant (see e.g. [23, p.3]) that the problem has a unique solution for each given set of function data $\rho_{ij} := f^{(j)}(\tau_i)/j!$. If not, the problem is termed *singular*. For example, if one specifies $f(0) = 0$, $f(1) = 0$, and $f'(1/2) = 0$, then *any* quadratic with zeros at 0 and 1 will have derivative zero half-way between; thus the solution is not unique. Contrariwise, specifying instead $f'(0) = 1$ (say) makes the problem have no solution. In both cases the problem is singular. Changing the node at $1/2$ to anything different, even $1/(2+\varepsilon)$ for ε small, makes the problem regular again. Thus in a certain sense the HB interpolation problem can be discontinuous, although in this instance the discontinuity is at least visible in the solution as $\varepsilon \to 0$:

$$f(t) = -\frac{(t-1)\,(2\,t + t\varepsilon + \varepsilon)\,f(0)}{\varepsilon} - \frac{t\,(t-1)\,(2+\varepsilon)\,f'(1/(2+\varepsilon))}{\varepsilon} + \frac{t\,(t\varepsilon + 2\,t - 2)\,f(1)}{\varepsilon} \tag{73}$$

and the solution visibly tends to infinity as $\varepsilon \to 0$. Continuity issues can be more complicated, however: see [10].

The HB interpolation problem can also be ‘structurally’ singular; for example, specifying a function value at 0 and a nonzero 2nd derivative, but not a 1st derivative, does not allow a linear polynomial to fit the data.

It is clear that a given HB interpolation problem is regular if and only if its associated matrix, obtained by deleting rows from the confluent Vandermonde matrix (rows corresponding to missing data) and deleting the same number of final columns (corresponding to the highest degree monomials), is square and has nonzero determinant.

Numerically, one expects that *near*-singularity will be important. If the structured condition number of the associated matrix is large, one would expect the resulting coefficients (in the monomial basis) to vary widely and erratically with perturbations in the data (and also, incidentally, with rounding errors in the computation).

We remark that the associated matrix is not used in the algorithm of this paper, although we will use it in a proof, momentarily. We do not express the HB interpolating polynomial in the monomial basis. Rather, we usually express it in a barycentric Hermite form. This avoids the potential ill-conditioning of the associated matrix.

However, we do not avoid ill-conditioning altogether (of course). If the HB interpolation problem is itself ill-conditioned, then the solution from our algorithm will reflect that, and accuracy will suffer. More, the $(m-1)$ by $(m-1)$ matrix (or the m by m matrix) that we *do* use in our algorithm may be ill-conditioned itself, possibly independently of the ill-conditioning of the underlying HB problem. We have not proved any results about the conditioning of this matrix, but remark that if $m \ll p$ then one would expect less difficulty by our method than by using the associated matrix, which is p by p.

We now discuss the relationship of the regularity of the problem with the success or failure of our algorithms.

5.1 The case $m = 1$

Let $H_{ik}(x)$, $1 \leqslant i \leqslant n$, $0 \leqslant k \leqslant s_i - 1$ be the Hermite interpolating basis on the nodes τ_i with confluencies s_i. Then it is known that

$$\frac{H_{ik}^{(j)}(\tau_\ell)}{j!} = \delta_{i\ell}\, \delta_{jk} \tag{74}$$

where $\delta_{i\ell}$ is the Kronecker delta function (the factorials are the matter of convenience). Hence the solution to the Hermite interpolation problem is

$$f(t) = \sum_{i=1}^{n} \sum_{j=0}^{s_i-1} \rho_{ij} H_{ij}(t)\,. \tag{75}$$

Now, we can (by interchanging orders of summation), give an explicit formula for these $H_{ij}(t)$ (which is not new but not widely known)

$$\begin{aligned}
\frac{f(t)}{w(t)} &= \sum_{i=1}^{n} \sum_{j=0}^{s_i-1} \sum_{k=0}^{j} \gamma_{ij} \rho_{ik} (t-\tau_i)^{k-j-1} \\
&= \sum_{i=1}^{n} \sum_{k=0}^{s_i-1} \sum_{j=k}^{s_i-1} \gamma_{ij} \rho_{ik} (t-\tau_i)^{k-j-1} \\
&= \sum_{i=1}^{n} \sum_{k=0}^{s_i-1} \sum_{\ell=0}^{s_i-1-k} \gamma_{i,\ell+k} \rho_{ik} (t-\tau_i)^{-\ell-1} \\
&= \sum_{i=1}^{n} \sum_{k=0}^{s_i-1} \rho_{ik} \left(\sum_{\ell=0}^{s_i-1-k} \gamma_{i,\ell+k} (t-\tau_i)^{-\ell-1} \right)
\end{aligned} \tag{76}$$

Hence

Lemma 5

$$H_{ik}(t) = w(t) \sum_{\ell=0}^{s_i-1-k} \gamma_{i.\ell+k} (t-\tau_i)^{-\ell-1}\,, \tag{77}$$

and

Lemma 6 *The leading coefficient of a basis element*

$$[t^p] H_{ik}(t) = \gamma_{i,k} \tag{78}$$

from the $\ell = 0$ *term of* (77).

Lemma 7 *Let* $\mathbf{V}$ *be the confluent Vandermonde matrix on* τ_i *with confluencies* s_i, $1 \leqslant i \leqslant n$. *Then, since the* τ_i *are distinct and so* $\mathbf{V}$ *is nonsingular, if we put*

$$\mathbf{A} = \begin{bmatrix} \mathbf{V} & \mathbf{R} \\ -\mathbf{T} & \mathbf{0} \end{bmatrix} \tag{79}$$

where

$$\mathbf{T} = \begin{bmatrix} 1 & t & \ldots & t^p \end{bmatrix} \tag{80}$$

and

$$\mathbf{R} = \begin{bmatrix} \rho_{10} \\ \rho_{11} \\ \vdots \\ \rho_{n,s_n-1} \end{bmatrix}, \tag{81}$$

then we have

$$\det \mathbf{A} = \det \mathbf{V} \cdot \det \left(0 + \mathbf{T}\mathbf{V}^{-1}\mathbf{R} \right) \tag{82}$$

by the Schur complement [18, Ch. 4]. Therefore,

$$\det \mathbf{A} = \det (\mathbf{V}) f(t) \tag{83}$$

because

$$\mathbf{V}^{-1}\mathbf{R} \tag{84}$$

gives the coefficients of f in the monomial basis.

Theorem 4 *The HB problem with a single missing datum (so $m = 1$) at say (I, J), is regular if and only if* $\det (\mathbf{V}) \neq 0$ *and the coefficient of $(z - \tau_I)^{-J-1}$, that is γ_{IJ}, is not* 0.

Proof Use Laplace expansion on $\det (\mathbf{A})$ beginning with the last column.

$$f(t) = \sum_{i=1}^{n} \sum_{j=0}^{s_i-1} \pm \rho_{ij} \det \left(\mathbf{V} \underset{(i,j)}{\longleftarrow} -\mathbf{T} \right) \tag{85}$$

where the notation

$$\mathbf{V} \underset{(i,j)}{\longleftarrow} -\mathbf{T} \tag{86}$$

means replace the row of $\mathbf{V}$ corresponding to the ith node, $j$the derivative with the row of monomials indicated.

But this shows

$$H_{ij}(t) = \pm \det \left(\mathbf{V} \underset{(i,j)}{\longleftarrow} -\mathbf{T} \right). \tag{87}$$

Now do the Laplace expansion of this $p \times p$ determinant to find the coefficient of t^p, which is $\pm \det (\mathbf{V}_B)$ where $\mathbf{V}_B$ is obtained from $\mathbf{V}$ by deleting the (i, j) row and the final column. This is precisely the HB $m = 1$ interpolating matrix; the problem is regular if and only if this determinant is not zero. Comparison gives

$$\pm \det (\mathbf{V}_B) = \det (\mathbf{V}) \gamma_{IJ} \tag{88}$$

and since $\det (\mathbf{V}) \neq 0$ the proof is complete.

Remark 5 We now are assured that if $\gamma_{IJ} = 0$, the HB problem missing only that datum is not regular. Previously, all we had claimed was that data if $\gamma_{IJ} \neq 0$ then the problem was regular, and the solution was given by our algorithm. Now we have proved that our algorithm is correct in the $m = 1$ case.

6 Concluding remarks

We have presented algorithms, of roughly quadratic cost, for solving the HB interpolation problem and related problems. We have provided MATLAB and MAPLE code implementing our algorithms. The method is not precisely new, in that contour integrals have been used for interpolation problems before, as have partial fractions. We believe, however, that it is new enough, and robust and efficient enough, to be potentially interesting. The flexibility of the algorithm (allowing solution of rational interpolation problems) is also attractive.

The detailed numerical stability properties of these algorithms remain open. We believe that the 'fill-in' method is more often stable than the 'semi-symbolic' method, based on our examples. We expect that for low-confluency problems, with small m, our algorithms may be preferential to existing algorithms.

Acknowledgements

This work was partially supported by the Natural Engineering Research Council of Canada. Special thanks go to the University of Newcastle, Australia, for an invitation that allowed RMC to visit JCB at a time when AS was also there. We also thank Piers Lawrence for writing the MATLAB implementation of the fill-in algorithm.

References

1. Elas Berriochoa and Alicia Cachafeiro. Algorithms for solving Hermite interpolation problems using the Fast Fourier Transform. *Journal of Computational and Applied Mathematics*, In Press, Corrected Proof:–, 2009.
2. Jean-Paul Berrut and Lloyd N. Trefethen. Barycentric Lagrange interpolation. *SIAM Review*, 46(3):501–517, 2004.
3. George David Birkhoff. General mean value and remainder theorems with applications to mechanical differentiation and quadrature. *Transactions of the American Mathematical Society*, 7(1):107–136, 1906.
4. R. W. Brankin and I. Gladwell. Shape-preserving local interpolation for plotting solutions of ODEs. *IMA Journal of Numerical Analysis*, 9:555–566, October 1989.
5. Manuel Bronstein and Bruno Salvy. Full partial fraction decomposition of rational functions. In *ISSAC '93: Proceedings of the 1993 international symposium on Symbolic and algebraic computation*, pages 157–160, New York, NY, USA, 1993. ACM.
6. John C. Butcher. A multistep generalization of Runge-Kutta methods with four or five stages. *J. ACM*, 14(1):84–99, 1967.
7. Francis Y. Chin. The partial fraction expansion problem and its inverse. *SIAM Journal on Computing*, 6(3):554–562, 1977.
8. Robert M. Corless, Azar Shakoori, Dhavide Aruliah, and Laureano Gonzalez-Vega. Barycentric Hermite interpolants for event location in initial-value problems. *Journal of Numerical Analysis, Industrial, and Applied Mathematics*, 3(1–2):1–18, 2008.
9. Robert M. Corless and Stephen M. Watt. Bernstein bases are optimal, but, sometimes, Lagrange bases are better. In *Proceedings of SYNASC, Timisoara*, pages 141–153. MIRTON Press, September 2004.
10. N. Dyn, G. G. Lorentz, and S. D. Riemenschneider. Continuity of the Birkhoff interpolation. *SIAM Journal on Numerical Analysis*, 19(3):507–509, 1982.
11. Joachim von zur Gathen and Jürgen Gerhard. *Modern computer algebra.* Cambridge University Press, Cambridge ; New York, 1999.
12. K. O. Geddes, S. R. Czapor, and G. Labahn. *Algorithms for computer algebra.* Kluwer Academic, Boston, 1992.
13. Peter Henrici. *Applied and computational complex analysis.* Wiley, New York, 1974; 1986.

14. Desmond J. Higham. Runge-Kutta defect control using Hermite-Birkhoff interpolation. *SIAM J. Sci. Stat. Comput.*, 12(5):991–999, 1991.
15. Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, second edition, 2002.
16. Nicholas J. Higham. The numerical stability of barycentric Lagrange interpolation. *IMA Journal of Numerical Analysis*, 24:547–556, 2004.
17. Nicholas J. Higham. *Functions of matrices : theory and computation.* Society for Industrial and Applied Mathematics, Philadelphia, 2008.
18. Leslie Hogben. *Handbook of linear algebra.* Chapman & Hall/CRC, Boca Raton, 2007. edited by Leslie Hogben ; associate editors, Richard Brualdi, Anne Greenbaum, Roy Mathias.; 1 v. (various pagings) : ill. ; 27 cm; Includes bibliographical references and indexes.
19. Shui-Hung Hou and Wan-Kai Pang. Inversion of confluent Vandermonde matrices. *Computers and Mathematics with Applications*, 43:1539–1547, 2002.
20. H. T. Kung and D. M. Tong. Fast algorithms for partial fraction decomposition. *SIAM Journal on Computing*, 6(3):582–593, 1977.
21. Sanjiva K. Lele. Compact finite difference schemes with spectral-like resolution. *Journal of Computational Physics*, 103(1):16 – 42, 1992.
22. Norman Levinson and Raymond M. Redheffer. *Complex variables.* Holden-Day, San Francisco, 1970.
23. G. G. Lorentz, K. Jetter, and S. D. Riemenschneider. *Birkhoff interpolation*, volume 19. Addison-Wesley Pub. Co. Advanced Book Program, World Science Div., Reading, Mass.; Don Mills, Ont., 1983.
24. Uwe Luther and Karla Rost. Matrix exponentials and inversion of confluent Vandermonde matrices. *Electronic Transactions on Numerical Analysis*, 18:91–100, 2004.
25. J. F. Mahoney and B. D. Sivazlian. Partial fractions expansion: a review of computational methodology and efficiency. *Journal of Computational and Applied Mathematics*, 9(3):247–269.
26. Louis M. Milne-Thomson. *The Calculus of Finite Diffrences.* Macmillan, London, 1933.
27. G. Mühlbach. An algorithmic approach to Hermite-Birkhoff interpolation. *Numerische Mathematik*, 37:339–347, 1981.
28. JiříFiala. An algorithm for Hermite-Birkhoff interpolation. *Applications of Mathematics*, 18(3):167–175, 1973.
29. C. Schneider and W. Werner. Hermite interpolation: the barycentric approach. *Computing*, 46:35–51, 1991.
30. David R. Stoutemyer. Multivariate partial fraction expansion. *ACM Commun. Comput. Algebra*, 42(4):206–210, 2008.
31. Ch. Tsitourras. Runge-Kutta interpolants for high-order precision computations. *Numerical Algorithms*, 44(3):291–307, 2007.
32. H. W. Turnbull. A note on partial fractions and determinants. *Proc. Edinburgh Math Society*, 1:49–54, 1927.
33. Jichao Zhao and Robert M. Corless. Compact finite difference method for integro-differential equations. *Appl. Math. Comput.*, 177(1):271–288, 2006.
34. Jichao Zhao, Matt Davison, and Robert M. Corless. Compact finite difference method for american option pricing. *J. Comput. Appl. Math.*, 206(1):306–321, 2007.