Numerische Mathematik
https://doi.org/10.1007/s00211-022-01316-w

Numerische Mathematik

On the numerical stability of linear barycentric rational interpolation

Chiara Fuda[1] · **Rosanna Campagna**[2] · **Kai Hormann**[1]

Received: 14 September 2021 / Revised: 27 June 2022 / Accepted: 1 September 2022

Abstract
The barycentric forms of polynomial and rational interpolation have recently gained popularity, because they can be computed with simple, efficient, and numerically stable algorithms. In this paper, we show more generally that the evaluation of any function that can be expressed as $r(x) = \sum_{i=0}^{n} a_i(x) f_i \big/ \sum_{j=0}^{m} b_j(x)$ in terms of data values f_i and some functions a_i and b_j for $i = 0, \ldots, n$ and $j = 0, \ldots, m$ with a simple algorithm that first sums up the terms in the numerator and the denominator, followed by a final division, is forward and backward stable under certain assumptions. This result includes the two barycentric forms of rational interpolation as special cases. Our analysis further reveals that the stability of the second barycentric form depends on the Lebesgue constant associated with the interpolation nodes, which typically grows with n, whereas the stability of the first barycentric form depends on a similar, but different quantity, that can be bounded in terms of the mesh ratio, regardless of n. We support our theoretical results with numerical experiments.

Mathematics Subject Classification 65D05 · 41A20 · 65G50

✉ Chiara Fuda
chiara.fuda@usi.ch

Rosanna Campagna
rosanna.campagna@unicampania.it

Kai Hormann
kai.hormann@usi.ch

[1] Faculty of Informatics, Università della Svizzera italiana, Via la Santa 1, 6962 Lugano, Switzerland

[2] Dipartimento di Matematica e Fisica, Università degli studi della Campania, Viale Abramo Lincoln 5, 81100 Caserta, Italy

Published online: 17 September 2022

1 Introduction

Given $n+1$ distinct interpolation nodes $x_0, \ldots, x_n$ with associated data $f_0, \ldots, f_n$, the classical interpolation problem consists in finding a function $r\colon \mathbb{R} \to \mathbb{R}$ that interpolates f_i at x_i, that is,

$$r(x_i) = f_i, \qquad i = 0, \ldots, n. \tag{1}$$

In the case of *polynomial* interpolation of degree at most n, this problem has a unique solution, which can be expressed in *Lagrange form* as

$$r(x) = \sum_{i=0}^{n} \ell_i(x) f_i, \qquad \ell_i(x) = \prod_{j=0,\, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}.$$

While this form is advantageous for theoretical analysis, its evaluation requires $O(n^2)$ operations and can be numerically unstable. It is advisable to consider instead the *first barycentric form* of r,

$$r(x) = \ell(x) \sum_{i=0}^{n} \frac{w_i}{x - x_i} f_i, \qquad \ell(x) = \prod_{j=0}^{n} (x - x_j), \tag{2}$$

with the *Lagrange weights* w_i defined as

$$w_i = \prod_{j=0,\, j\neq i}^{n} \frac{1}{x_i - x_j}, \qquad i = 0, \ldots, n. \tag{3}$$

The first barycentric form is more efficient than the Lagrange form, as it can be evaluated in $O(n)$ operations, after computing the w_i, which are independent of x, in $O(n^2)$ operations in a preprocessing step. Higham [1] further shows that this evaluation is backward stable with respect to perturbations of the data f_i. Another means of evaluating r is given by the *second barycentric form* of r,

$$r(x) = \frac{\sum_{i=0}^{n} \frac{w_i}{x - x_i} f_i}{\sum_{i=0}^{n} \frac{w_i}{x - x_i}}, \tag{4}$$

which can be derived from (2) by noticing that $1 = \ell(x) \sum_{i=0}^{n} \frac{w_i}{x - x_i}$. Evaluating this formula also requires $O(n)$ operations, but it comes with the advantage that the w_i can be rescaled by a common factor to avoid underflow and overflow [2]. Moreover, the second barycentric form is forward stable, as long as the Lebesgue constant associated with the interpolation nodes x_i is small [1], which is the case, for example, for Chebyshev nodes of the first and the second kind, but not for equidistant nodes.

In the case of *rational* interpolation, the interpolation problem (1) no longer has a unique solution, but Berrut and Mittelmann [3] show that every rational interpolant

of degree at most n can be expressed in the second barycentric form (4) for a specific choice of weights w_i. Vice versa, Schneider and Werner [4] note that for any set of non-zero weights w_i, the function r in (4) is a rational interpolant of degree at most n. An important subset of these *barycentric rational interpolants* are those that do not have any poles in $\mathbb{R}$. This is obviously true for the Lagrange weights in (3), but also for the *Berrut weights* [5]

$$w_i = (-1)^i, \qquad i = 0, \ldots, n, \tag{5}$$

and, assuming the interpolation nodes to be in ascending order, that is, $x_0 < \cdots < x_n$, for the family of weights with parameter $d \in \{0, \ldots, n\}$,

$$w_i = \sum_{k=\max(i-d,0)}^{\min(i,n-d)} (-1)^k \prod_{j=k,\, j\neq i}^{k+d} \frac{1}{x_i - x_j}, \qquad i = 0, \ldots, n, \tag{6}$$

proposed by Floater and Hormann [6]. Note that this family includes the Berrut and the Lagrange weights as special cases for $d = 0$ and $d = n$. For the weights in (6), Floater and Hormann further observe that the barycentric rational interpolant in (4) with the weights (6) can also be written as

$$r(x) = \frac{\sum_{i=0}^{n} \frac{w_i}{x - x_i} f_i}{\sum_{i=0}^{n-d} \lambda_i(x)}, \tag{7}$$

where

$$\lambda_i(x) = \frac{(-1)^i}{(x - x_i) \cdots (x - x_{i+d})}, \qquad i = 0, \ldots, n - d. \tag{8}$$

As the formula in (7) simplifies to (2) for $d = n$, we refer to it as the *first barycentric form* of the corresponding rational interpolant. Note that the first and the second form are identical for Berrut's interpolant, that is, for $d = 0$.

The result of Higham [1] can be extended to show that the evaluation of the second barycentric form (4) is forward stable, not only in the case of polynomial interpolation, but for general barycentric rational interpolants, provided that the weights w_i can be computed with a forward stable algorithm and that the corresponding Lebesgue constant is small [7]. For Berrut's rational interpolant with weights in (5), this is the case for all well-spaced interpolation nodes [8], including equidistant and Chebyshev nodes. For the family of barycentric rational interpolants with weights in (6), the Lebesgue constant is known to grow logarithmically in n, for any constant $d > 0$ and equidistant [9] as well as quasi-equidistant [10] nodes, and the formula in (6) turns out to be a forward stable means of computing the weights w_i [11].

In this paper, we further generalize the proof of Salazar Celis [7], such that it can also be used for proving the forward stability of the first barycentric form (7), with two important changes (Sect. 3). On the one hand, the result relies on the fact that not only the w_i, but also the λ_i can be evaluated with a forward stable algorithm. This, however is indeed the case, both for the formula in (8), which requires $O(d(n - d))$

operations for $0 < d < n$, as well as a more efficient, but slightly less stable formula that gets by with $O(n)$ operations (Sect. 4). On the other hand, the Lebesgue constant must be replaced by a similar quantity that depends on the functions λ_i, for which we prove that it is at most on the order of $O(\mu^d)$, where μ is the *mesh ratio* of the interpolation nodes (Sect. 6). Moreover, we show that a more efficient formula [12] for computing the weights in (6) is forward stable, too (Sect. 4).

Regarding backward stability, Mascarenhas and Camargo [13] show that the second barycentric form is backward stable under the same assumptions, namely forward stable weights w_i and small Lebesgue constant. Moreover, Camargo [11] proves that the first barycentric form is backward stable, as long as the denominator in (7) is computed with a special algorithm in $O(nd)$ operations.

In this paper, we derive a substantially different approach that can be used to show the backward stability of both barycentric forms (Sect. 5). For the second barycentric form, our result provides an upper bound on the perturbation of the data f_i that is smaller than the upper bound by Mascarenhas and Camargo [13]. For the first barycentric form, our upper bound is larger than the one found by Camargo [11], but it comes with the advantage of holding for a more efficient way of computing the denominator in (7) in $O(n)$ operations, which is based on our new $O(n)$ algorithm for evaluating the λ_i (Sect. 4).

It is important to note that our results hold under the assumption that the input values to the algorithm, x_i, f_i, and x, are given as floating point numbers, so they do not introduce any additional error when we compute both forms (4) and (7). Our stability analysis does not cover errors that result from initially rounding the given values to floating point numbers.

Our numerical experiments (Sect. 7) confirm that this leads to an evaluation of the first barycentric form (7) of the rational interpolant with weights in (6), which is considerably faster than the algorithm proposed by Camargo [11], especially for larger d, at the price of only marginally larger forward errors. Evaluating the interpolant using the second barycentric form (4) is even faster and can be as stable, but may also result in significantly larger errors for certain choices of interpolation nodes. However, we also report a case in which the second form is stable and the first is not.

2 Preliminaries

Following Trefethen [14], a general *problem* g can be seen as a function $g\colon U \to V$ from a normed vector space $(U, \|\cdot\|_U)$ of *data* to a normed vector space $(V, \|\cdot\|_V)$ of *solutions* and a *numerical algorithm* for solving the problem on a computer as another function $\hat{g}\colon U \to V$ that approximates g.

For our analysis, we consider a computer that uses a set $\mathbb{F}$ of *floating point numbers* with the corresponding *machine epsilon* ϵ and let $\mathrm{fl}\colon \mathbb{R} \to \mathbb{F}$ be the *rounding function* that maps each $x \in \mathbb{R}$ to the closest floating point approximation $\mathrm{fl}(x) \in \mathbb{F}$. Moreover, we denote by $\circledast$ the floating point analogue of the arithmetic operation $* \in \{+, -, \times, \div\}$, that is,

$$x \circledast y = \mathrm{fl}(x * y) \tag{9}$$

for $x, y \in \mathbb{F}$, and recall [14, Lecture 13] that fl and $\circledast$ introduce a *relative error* of size at most ϵ. That is, for all $x \in \mathbb{R}$ there exists some $\delta \in \mathbb{R}$ with $|\delta| < \epsilon$, such that

$$\mathrm{fl}(x) = x(1+\delta)$$

and likewise

$$x \circledast y = (x * y)(1+\delta), \qquad * \in \{+, -, \times, \div\} \tag{10}$$

for all $x, y \in \mathbb{F}$, which follows immediately from (9). We further note that $\mathrm{fl}(-x) = -x$ for all $x \in \mathbb{F}$, so that multiplying a floating point number by $(-1)^i$ or taking its absolute value does not entail any rounding error.

In order to analyse and to compare the quality of numerical algorithms, the usual quantities to consider are the *absolute forward error* $\|\hat{g}(u) - g(u)\|_V$ and the *relative forward error* $\|\hat{g}(u) - g(u)\|_V / \|g(u)\|_V$. The algorithm $\hat{g}$ is called *accurate* or *forward stable*, if

$$\frac{\|\hat{g}(u) - g(u)\|_V}{\|g(u)\|_V} = O(\epsilon)$$

for all $u \in U$ with $g(u) \neq 0$ and *backward stable* at $u \neq 0$, if

$$\hat{g}(u) = g(\hat{u}) \qquad \text{for some} \quad \hat{u} \in U \quad \text{with} \quad \frac{\|\hat{u} - u\|_U}{\|u\|_U} = O(\epsilon),$$

where the notation $x = O(\epsilon)$ means that there exists some positive constant C, such that $|x| \leq C\epsilon$ as $\epsilon \to 0$. In the case of forward stability, this constant usually depends on the *relative condition number*

$$\kappa(u) = \lim_{\delta \to 0} \sup_{\|h\|_U \leq \delta} \left(\frac{\|g(u+h) - g(u)\|_V}{\|g(u)\|_V} \Big/ \frac{\|h\|_U}{\|u\|_U} \right)$$

of the problem g at u and on the dimensions of U and V.

In this paper, we analyse algorithms for evaluating barycentric rational interpolants, based on the formulas (4) and (7). Following Higham [1], we assume the evaluation point $x \in \mathbb{F}$ and the interpolation nodes $\boldsymbol{x} = (x_0, \ldots, x_n) \in \mathbb{F}^{n+1}$ to be fixed and prove the forward and backward stability with respect to the data $\boldsymbol{f} = (f_0, \ldots, f_n) \in \mathbb{F}^{n+1}$. More precisely, we solve the problem

$$g \colon (\mathbb{F}^{n+1}, \|\cdot\|_\infty) \to (\mathbb{R}, |\cdot|), \qquad g(\boldsymbol{f}) = r(x)$$

by a numerical algorithm

$$\hat{g} \colon (\mathbb{F}^{n+1}, \|\cdot\|_\infty) \to (\mathbb{F}, |\cdot|), \qquad \hat{g}(\boldsymbol{f}) = \hat{r}(x),$$

and we show that for any choice of x, $\boldsymbol{x}$, and $\boldsymbol{f}$, the relative forward error is bounded from above (Sect. 3),

$$\frac{|\hat{r}(x) - r(x)|}{|r(x)|} \leq C_1\epsilon + O(\epsilon^2),$$

and that there exists some perturbed data $\hat{f}$ with

$$\frac{\|\hat{f} - \boldsymbol{f}\|_\infty}{\|\boldsymbol{f}\|_\infty} \leq C_2\epsilon + O(\epsilon^2),$$

such that $\hat{r}(x; \boldsymbol{x}, \boldsymbol{f}) = r(x; \boldsymbol{x}, \hat{f})$, where the additional arguments of r are used to emphasize the dependence of the interpolant on the interpolation nodes and the data (Sect. 5). In both cases, ϵ is assumed to be sufficiently small, since this is enough to guarantee that the errors are $O(\epsilon)$. We will see that both constants C_1 and C_2 depend linearly on n and a quantity $\beta(x)$, which is independent of $\boldsymbol{f}$ and different for the first and the second barycentric form. It turns out that this quantity is usually smaller when the first barycentric form is used to evaluate r (Sect. 7). As mentioned above, the constant C_1 further depends on the condition number of the problem, and more specifically on the *componentwise relative condition number* [15]

$$\kappa(x; \boldsymbol{x}, \boldsymbol{f}) = \lim_{\delta \to 0} \sup_{\substack{\boldsymbol{h} \in \mathbb{R}^{n+1} \setminus \{\boldsymbol{0}\} \\ |h_i| \leq \delta |f_i|}} \left(\frac{|r(x; \boldsymbol{x}, \boldsymbol{f} + \boldsymbol{h}) - r(x; \boldsymbol{x}, \boldsymbol{f})|}{|r(x; \boldsymbol{x}, \boldsymbol{f})|} \Bigg/ \max_{\substack{i=0,\ldots,n \\ f_i \neq 0}} \frac{|h_i|}{|f_i|} \right). \quad (11)$$

For the derivation of these upper bounds, we shall frequently use the following basic facts:

1. By Taylor expansion,

$$\frac{1}{1+y} = \sum_{k=0}^{\infty} (-1)^k y^k$$

for any $y \in \mathbb{R}$ with $|y| < 1$. Consequently, if $y = O(\epsilon)$, then

$$\frac{1}{1+y} = 1 - y + O(\epsilon^2). \quad (12)$$

Moreover, for any $\delta \in \mathbb{R}$ with $|\delta| \leq \epsilon$, there exists some $\delta' \in \mathbb{R}$ with $|\delta'| \leq \epsilon + O(\epsilon^2)$, such that

$$\frac{1}{1+\delta} = 1 + \delta'. \quad (13)$$

This observation is useful for "moving" the perturbation (10) caused by a floating point operation from the denominator to the numerator.

2. For any $\delta_1, \ldots, \delta_m \in \mathbb{R}$ with $|\delta_i| \leq C_i\epsilon$ for some $C_i > 0$, $i = 0, \ldots, n$, there exists some $\delta \in \mathbb{R}$ with $|\delta| \leq C\epsilon + O(\epsilon^2)$, where $C = \sum_{i=1}^{m} C_i$, such that

$$\prod_{j=1}^{m}(1+\delta_j) = 1 + \delta. \tag{14}$$

We use this observation to gather the perturbations caused by computing the product of m terms into a single perturbation[1].

3. For any $t_0, \ldots, t_m \in \mathbb{F}$, there exist some $\varphi_0, \ldots, \varphi_m \in \mathbb{R}$ with $|\varphi_0|, \ldots, |\varphi_m| \leq m\epsilon + O(\epsilon^2)$, such that

$$\mathrm{fl}\left(\sum_{i=0}^{m} t_i\right) = (\cdots((t_0 \oplus t_1) \oplus t_2)\cdots \oplus t_m) = \sum_{i=0}^{m} t_i(1+\varphi_i). \tag{15}$$

This follows from the previous observation, and we use it to estimate the rounding error introduced by simple iterative summation of $m+1$ floating point numbers.

3 Forward stability

For analysing the relative forward error of barycentric rational interpolation, we first observe that (4) and (7) can both be written in the common form

$$r(x) = \frac{\sum_{i=0}^{n} a_i(x) f_i}{\sum_{j=0}^{m} b_j(x)}, \tag{16}$$

where $a_i(x) = w_i/(x - x_i)$ for both forms, while $m = n$ and $b_j(x) = a_j(x)$ for the second form and $m = n - d$ and $b_j(x) = \lambda_j(x)$ for the first form. Next, we define the functions

$$\alpha(x; \boldsymbol{f}) = \frac{\sum_{i=0}^{n} |a_i(x) f_i|}{|\sum_{i=0}^{n} a_i(x) f_i|} \tag{17}$$

and

$$\beta(x) = \frac{\sum_{j=0}^{m} |b_j(x)|}{|\sum_{j=0}^{m} b_j(x)|}. \tag{18}$$

Assuming now that we have forward stable algorithms for computing $a_i(x)$ as $\hat{a}_i(x)$ and $b_j(x)$ as $\hat{b}_j(x)$, we can derive a general bound on the relative forward error for the function r in (16).

Theorem 1 *Suppose that there exist* $\alpha_0, \ldots, \alpha_n \in \mathbb{R}$ *with*

$$\hat{a}_i(x) = a_i(x)(1+\alpha_i), \qquad |\alpha_i| \leq A\epsilon + O(\epsilon^2), \qquad i = 0, \ldots, n$$

[1] This is somewhat similar to counting the number of floating point operations with Stewart's relative error counter [16] and sufficient for our needs.

and $\beta_0, \ldots, \beta_m \in \mathbb{R}$ *with*

$$\hat{b}_j(x) = b_j(x)(1+\beta_j), \qquad |\beta_j| \le B\epsilon + O(\epsilon^2), \qquad j = 0, \ldots, m$$

for some constants A and B. Then, assuming $\boldsymbol{f} \in \mathbb{F}^{n+1}$*, the relative forward error of r in* (16) *satisfies*

$$\frac{|\hat{r}(x) - r(x)|}{|r(x)|} \le (n+2+A)\alpha(x;\boldsymbol{f})\epsilon + (m+B)\beta(x)\epsilon + O(\epsilon^2), \tag{19}$$

for ϵ *small enough.*

Proof We first notice that $\hat{r}$ is given by

$$\begin{aligned}\hat{r}(x) &= \frac{\sum_{i=0}^n \hat{a}_i(x) f_i (1+\delta_i^\times)(1+\varphi_i^N)}{\sum_{j=0}^m \hat{b}_j(x)(1+\varphi_j^D)}(1+\delta^\div) \\ &= \frac{\sum_{i=0}^n a_i(x)(1+\alpha_i) f_i (1+\delta_i^\times)(1+\varphi_i^N)}{\sum_{j=0}^m b_j(x)(1+\beta_j)(1+\varphi_j^D)}(1+\delta^\div),\end{aligned}$$

where $\delta_i^\times, \varphi_i^N, \varphi_j^D$ and $\delta^\div$ are the relative errors introduced respectively by the product $\hat{a}_i(x) f_i$, the sums in the numerator and the denominator, and the final division. It then follows from (10) that $|\delta_i^\times|, |\delta^\div| \le \epsilon$, while from (15) we have $|\varphi_i^N| \le n\epsilon + O(\epsilon^2)$ and $|\varphi_j^D| \le m\epsilon + O(\epsilon^2)$. By (14), there exist some $\eta_i, \mu_j \in \mathbb{R}$ with

$$\begin{aligned}|\eta_i| &\le (n+2+A)\epsilon + O(\epsilon^2), \qquad & i = 0, \ldots, n, \\ |\mu_j| &\le (m+B)\epsilon + O(\epsilon^2), \qquad & j = 0, \ldots, m,\end{aligned} \tag{20}$$

such that

$$\hat{r}(x) = \frac{\sum_{i=0}^n a_i(x) f_i (1+\eta_i)}{\sum_{j=0}^m b_j(x)(1+\mu_j)}. \tag{21}$$

Therefore,

$$\begin{aligned}\frac{\hat{r}(x)}{r(x)} &= \frac{\sum_{i=0}^n a_i(x) f_i (1+\eta_i)}{\sum_{j=0}^m b_j(x)(1+\mu_j)} \Bigg/ \frac{\sum_{i=0}^n a_i(x) f_i}{\sum_{j=0}^m b_j(x)} \\ &= \frac{\sum_{i=0}^n a_i(x) f_i (1+\eta_i)}{\sum_{j=0}^m a_i(x) f_i} \frac{\sum_{j=0}^m b_j(x)}{\sum_{j=0}^m b_j(x)(1+\mu_j)} \\ &= \left(1 + \frac{\sum_{i=0}^n a_i(x) f_i \eta_i}{\sum_{j=0}^m a_i(x) f_i}\right) \frac{1}{1 + \frac{\sum_{j=0}^m b_j(x)\mu_j}{\sum_{j=0}^m b_j(x)}}.\end{aligned}$$

Since, by the triangle inequality,

$$\left|\frac{\sum_{i=0}^n a_i(x)\eta_i}{\sum_{i=0}^n a_i(x)}\right| \le \alpha(x) \max_{i=0,\ldots,n} |\eta_i| \le \alpha(x)(n+2+A)\epsilon + O(\epsilon^2) \tag{22}$$

and

$$\left|\frac{\sum_{j=0}^{m} b_j(x)\mu_j}{\sum_{j=0}^{m} b_j(x)}\right| \leq \beta(x) \max_{j=0,\ldots,m} |\mu_j| \leq \beta(x)(m+B)\epsilon + O(\epsilon^2), \tag{23}$$

we can use (12) to express this ratio as

$$\begin{aligned}\frac{\hat{r}(x)}{r(x)} &= \left(1 + \frac{\sum_{i=0}^{n} a_i(x) f_i \eta_i}{\sum_{j=0}^{m} a_i(x) f_i}\right)\left(1 - \frac{\sum_{j=0}^{m} b_j(x)\mu_j}{\sum_{j=0}^{m} b_j(x)} + O(\epsilon^2)\right) \\ &= 1 + \frac{\sum_{i=0}^{n} a_i(x) f_i \eta_i}{\sum_{j=0}^{m} a_i(x) f_i} - \frac{\sum_{j=0}^{m} b_j(x)\mu_j}{\sum_{j=0}^{m} b_j(x)} + O(\epsilon^2)\end{aligned}$$

and obtain the relative forward error of r as

$$\frac{|\hat{r}(x) - r(x)|}{|r(x)|} = \left|\frac{\hat{r}(x)}{r(x)} - 1\right| = \left|\frac{\sum_{i=0}^{n} a_i(x) f_i \eta_i}{\sum_{j=0}^{m} a_i(x) f_i} - \frac{\sum_{j=0}^{m} b_j(x)\mu_j}{\sum_{j=0}^{m} b_j(x)} + O(\epsilon^2)\right|.$$

The upper bound in (19) then follows immediately by using again (22) and (23). □

While Theorem 1 holds for any function r that can be expressed as in (16), we shall now focus on the special cases for the two different forms of the barycentric rational interpolant. For the second barycentric form, the only assumption we need is that the weights w_i can be computed as $\hat{w}_i$ with a forward stable algorithm. Moreover, we recall the definition of the *Lebesgue function* of the barycentric rational interpolant [9] as

$$\Lambda_n(x; \boldsymbol{x}) = \frac{\sum_{i=0}^{n} \left|\frac{w_i}{x - x_i}\right|}{\left|\sum_{i=0}^{n} \frac{w_i}{x - x_i}\right|}.$$

Corollary 1 *Assume that there exist* $\psi_0, \ldots, \psi_n \in \mathbb{R}$ *with*

$$\hat{w}_i = w_i(1 + \psi_i), \qquad |\psi_i| \leq W\epsilon + O(\epsilon^2), \qquad i = 0, \ldots, n \tag{24}$$

for some constant W*. Then, assuming* $\boldsymbol{f} \in \mathbb{F}^{n+1}$*, the relative forward error of the second barycentric form in* (4) *satisfies*

$$\frac{|\hat{r}(x) - r(x)|}{|r(x)|} \leq (n + 4 + W)\kappa(x; \boldsymbol{x}, \boldsymbol{f})\epsilon + (n + 2 + W)\Lambda_n(x; \boldsymbol{x})\epsilon + O(\epsilon^2), \tag{25}$$

for ϵ *small enough.*

Proof We first notice that $a_i(x) = w_i/(x - x_i)$ can be computed with one subtraction and one division, so that, by (10), (13), (14), and (24),

$$\hat{a}_i(x) = a_i(x)(1 + \alpha_i), \qquad i = 0, \ldots, n$$

for some $\alpha_i \in \mathbb{R}$ with $|\alpha_i| \leq (2+W)\epsilon + O(\epsilon^2)$. Hence, the constants in (19) are $A = 2 + W$ and further $B = 2 + W$, because $b_j(x) = a_j(x)$ in case of the second barycentric form. From the latter, it also follows immediately that $\beta(x)$ in (18) is equal to $\Lambda_n(x; \boldsymbol{x})$ in this case, and it only remains to show that $\alpha(x; \boldsymbol{f})$ in (17) is equal to $\kappa(x; \boldsymbol{x}, \boldsymbol{f})$ in (11). To this end, we first use the triangle inequality to see that for any $\delta > 0$ and any $\boldsymbol{h} = (h_0, \ldots, h_n) \in \mathbb{R}^{n+1} \setminus \{\boldsymbol{0}\}$ with $|h_i| \leq \delta |f_i|$,

$$\left|\sum_{i=0}^{n} a_i(x) h_i\right| = \left|\sum_{i=0}^{n} a_i(x) f_i \frac{h_i}{f_i}\right| \leq H \sum_{i=0}^{n} |a_i(x) f_i|, \qquad H = \max_{\substack{i=0,\ldots,n \\ f_i \neq 0}} \frac{|h_i|}{|f_i|},$$

where equality is attained for $\boldsymbol{h}$ with $h_i = \delta \operatorname{sign}(a_i(x))|f_i|$, $i = 0, \ldots, n$. Dividing both sides of this inequality by H and $\left|\sum_{i=0}^{n} a_i(x) f_i\right|$, taking the supremum over all admissible $\boldsymbol{h}$ and the limit $\delta \to 0$, gives $\kappa(x; \boldsymbol{x}, \boldsymbol{f}) = \alpha(x; \boldsymbol{f})$. □

For the first barycentric form, we additionally need to assume that the $\lambda_i(x)$ can be computed with a forward stable algorithm as $\hat{\lambda}_i(x)$, and we note that $\beta(x)$ in (18) is equal to

$$\Gamma_d(x; \boldsymbol{x}) = \frac{\sum_{i=0}^{n-d} |\lambda_i(x)|}{\left|\sum_{i=0}^{n-d} \lambda_i(x)\right|}. \tag{26}$$

in this case.

Corollary 2 *Assume that the weights $w_0, \ldots, w_n$ can be computed as in (24) and that there exist $\gamma_0, \ldots, \gamma_{n-d} \in \mathbb{R}$ with*

$$\hat{\lambda}_i(x) = \lambda_i(x)(1 + \gamma_i), \qquad |\gamma_i| \leq C\epsilon + O(\epsilon^2), \qquad i = 0, \ldots, n-d \tag{27}$$

for some constant C. Then, assuming $\boldsymbol{f} \in \mathbb{F}^{n+1}$, the relative forward error of the first barycentric form in (7) satisfies

$$\frac{|\hat{r}(x) - r(x)|}{|r(x)|} \leq (n + 4 + W)\kappa(x; \boldsymbol{x}, \boldsymbol{f})\epsilon + (n - d + C)\Gamma_d(x; \boldsymbol{x})\epsilon + O(\epsilon^2), \tag{28}$$

for ϵ small enough.

Proof As the numerator of the first and second barycentric form are identical, the only difference to the proof of Corollary 1 is that $B = C$ in (19), because $b_j(x) = \lambda_j(x)$ and $m = n - d$. □

While upper bounds for the Lebesgue function $\Lambda_n(x; \boldsymbol{x})$ can be found in the literature [8, 9], we are unaware of any previous work bounding the function $\Gamma_d(x; \boldsymbol{x})$, and we derive such an upper bound in Sect. 6.

4 Computing the weights w_i and evaluating the functions λ_i

It remains to work out the constants W and C, related to the computation of the weights w_i in (4) and the evaluation of the functions λ_i in (7). In particular, we study

the error propagation that occurs in the implementation of different algorithms and further analyse them in terms of computational cost.

Regarding the w_i, it was shown by Higham [1] that the Lagrange weights in (3) can be computed stably with $W = 2n$ in (24), and the Berrut weights in (5) can be represented exactly in $\mathbb{F}$, so that $W = 0$. The same holds for the weights in (6) if the interpolation nodes are equidistant, because they simplify to the integers

$$w_i = (-1)^{i-d} \sum_{j=\max(i-d,0)}^{\min(i,n-d)} \binom{d}{i-j}, \qquad i = 0, \ldots, n$$

in this special case [6]. For the general case, Camargo [11, Lemma 1] shows that $W = 3d$, if the w_i are computed with a straightforward implementation of the formula in (6). While this construction requires $O(nd^2)$ operations, Hormann and Schaefer [12] suggest an improved $O(nd)$ pyramid algorithm, which turns out to have the same precision. Their algorithm starts from the values

$$v_i^d = 1, \qquad i = 0, \ldots, n-d \tag{29}$$

and iteratively computes

$$v_i^l = \frac{v_{i-1}^{l+1}}{x_{i+l} - x_{i-1}} + \frac{v_i^{l+1}}{x_{i+l+1} - x_i}, \qquad i = 0, \ldots, n-l \tag{30}$$

for $l = d-1, d-2, \ldots, 0$, tacitly assuming $v_i^l = 0$ for $i < 0$ and $i > n-l$. They show that the resulting values v_i^0 are essentially the weights w_i in (6), up to a factor of $(-1)^{i-d}$.

Lemma 1 *For any $x_0, \ldots, x_n \in \mathbb{F}$, there exist $\phi_0^0, \ldots, \phi_n^0 \in \mathbb{R}$ with $|\phi_0^0|, \ldots, |\phi_n^0| \leq W\epsilon + O(\epsilon^2)$ for $W = 3d$, such that the v_i^0 in (30) satisfy*

$$\hat{v}_i^0 = v_i^0(1 + \phi_i^0), \qquad i = 0, \ldots, n.$$

Proof The statement is a special case of the more general observation that there exists for any $l = d, d-1, \ldots, 0$ and $i = 0, \ldots, n-l$ some $\phi_i^l \in \mathbb{R}$ with $|\phi_i^l| \leq 3(d-l)\epsilon + O(\epsilon^2)$, such that $\hat{v}_i^l = v_i^l(1 + \phi_i^l)$, which can be shown by induction over l. The base case $l = d$ follows trivially from (29). For the inductive step from $l+1$ to l, we conclude from (10), (13), and (14), that $\hat{v}_i^l$, computed with the formula in (30), satisfies

$$\hat{v}_i^l = \frac{\hat{v}_{i-1}^{l+1}}{x_{i+l} - x_{i-1}}(1 + \rho_1) + \frac{\hat{v}_i^{l+1}}{x_{i+l+1} - x_i}(1 + \rho_2)$$

for some $\rho_1, \rho_2 \in \mathbb{R}$ with $|\rho_1|, |\rho_2| \leq 3\epsilon + O(\epsilon^2)$, since both terms are affected by one subtraction, one division, and one sum. By induction hypothesis and (14), we can

then assume the existence of some $\sigma_1, \sigma_2 \in \mathbb{R}$ with $|\sigma_1|, |\sigma_2| \leq 3(d-l)\epsilon + O(\epsilon^2)$, such that

$$\hat{v}_i^l = \frac{v_{i-1}^{l+1}}{x_{i+l} - x_{i-1}}(1+\sigma_1) + \frac{v_i^{l+1}}{x_{i+l+1} - x_i}(1+\sigma_2),$$

and the intermediate value theorem further guarantees that

$$\hat{v}_i^l = \left(\frac{v_{i-1}^{l+1}}{x_{i+l} - x_{i-1}} + \frac{v_i^{l+1}}{x_{i+l+1} - x_i}\right)(1+\phi_i^l)$$

for some $\phi_i^l \in [\min(\sigma_1, \sigma_2), \max(\sigma_1, \sigma_2)]$ with $|\phi_i^l| \leq 3(d-l)\epsilon + O(\epsilon^2)$. □

Let us now focus on the functions λ_i that appear in the barycentric formula (7) and first study the error propagation when computing them straightforwardly, following the formula in (8).

Lemma 2 *For any $x \in \mathbb{F}$ and $x_0, \ldots, x_n \in \mathbb{F}$, there exist $\theta_0, \ldots, \theta_{n-d} \in \mathbb{R}$ with $|\theta_0|, \ldots, |\theta_{n-d}| \leq C\epsilon + O(\epsilon^2)$ for $C = 2d+2$, such that the $\lambda_i(x)$ in (8) satisfy*

$$\hat{\lambda}_i(x) = \lambda_i(x)(1+\theta_i), \qquad i = 0, \ldots, n-d.$$

Proof Since computing $\hat{\lambda}_i(x)$ requires $d+1$ subtractions, d products, and one division, the result follows directly from (10), (14), and (13). □

Evaluating λ_i in this way clearly has a computational cost of $O(d)$ for $d > 0$ and $O(1)$ for $d = 0$, so that computing all $\lambda_i(x)$ requires $O(d(n-d))$ operations for $0 < d < n$ and $O(n)$ operations for $d = 0$ and $d = n$. However, for $0 < d < n$ this can be improved by exploiting the fact that $\lambda_i(x)$ and $\lambda_{i+1}(x)$ have d common factors in the denominator, which in turn suggests to first compute the "central" $\lambda_m(x)$ for $m = \lfloor \frac{n-d}{2} \rfloor$ as above and then the remaining $\lambda_i(x)$ iteratively as

$$\begin{aligned} \lambda_{i-1}(x) &= -\lambda_i(x)\frac{(x - x_{i+d})}{(x - x_{i-1})}, && i = m, m-1, \ldots, 1, \\ \lambda_{i+1}(x) &= -\lambda_i(x)\frac{(x - x_i)}{(x - x_{i+1+d})}, && i = m, m+1, \ldots, n-d-1. \end{aligned} \tag{31}$$

Computing all $\lambda_i(x)$ this way requires only $O(n)$ operations, but it comes at the price of a likely reduced precision.

Lemma 3 *For any $x \in \mathbb{F}$ and $x_0, \ldots, x_n \in \mathbb{F}$, there exist $\zeta_0, \ldots, \zeta_{n-d} \in \mathbb{R}$ with $|\zeta_0|, \ldots, |\zeta_{n-d}| \leq C\epsilon + O(\epsilon^2)$ for $C = 2n+4$, such that the $\lambda_i(x)$ in (31) satisfy*

$$\hat{\lambda}_i(x) = \lambda_i(x)(1+\zeta_i), \qquad i = 0, \ldots, n-d.$$

Proof By Lemma 2, we know that $\hat{\lambda}_m(x) = \lambda_m(x)(1+\zeta_m)$ with $|\zeta_m| \leq (2d+2)\epsilon + O(\epsilon^2)$. Each step in (31) involves two subtractions, one division, and one product, and therefore introduces a perturbation of $(1+\delta)$ with $|\delta| \leq 4\epsilon + O(\epsilon^2)$, and these perturbations accumulate during the iteration. Since the number of steps is at most $(n-d+1)/2$, the overall perturbation for each $\lambda_i(x)$ is at most $(1+\zeta_i)$ with $|\zeta_i| \leq [(2d+2) + 4(n-d+1)/2]\epsilon + O(\epsilon^2)$. □

5 Backward stability

Similar to how we established the forward stability of both barycentric forms in a unified way in Sect. 3, we can prove the backward stability in general for the function r in (16) and then derive upper bounds on the perturbation of the data for both forms as special cases.

Theorem 2 *Suppose that there exist* $\alpha_0, \ldots, \alpha_n \in \mathbb{R}$ *with*

$$\hat{a}_i(x) = a_i(x)(1+\alpha_i), \qquad |\alpha_i| \leq A\epsilon + O(\epsilon^2), \qquad i = 0, \ldots, n$$

and $\beta_0, \ldots, \beta_m \in \mathbb{R}$ *with*

$$\hat{b}_j(x) = b_j(x)(1+\beta_j), \qquad |\beta_j| \leq B\epsilon + O(\epsilon^2), \qquad j = 0, \ldots, m$$

for some constants A and B. Then there exists for any $\boldsymbol{f} \in \mathbb{F}^{n+1}$ *some* $\hat{f} \in \mathbb{F}^{n+1}$ *with*

$$\frac{\|\hat{f} - \boldsymbol{f}\|_\infty}{\|\boldsymbol{f}\|_\infty} \leq (n+2+A)\epsilon + (m+B)\max_x \beta(x)\epsilon + O(\epsilon^2),$$

for ϵ *small enough, such that the numerical evaluation of* r *in* (16) *satisfies* $\hat{r}(x;\boldsymbol{f}) = r(x;\hat{f})$.

Proof Starting from (21), with the η_i and μ_j satisfying (20), we get, again with the help of (12),

$$\begin{aligned}
\hat{r}(x;\boldsymbol{x},\boldsymbol{f}) &= \frac{\sum_{i=0}^n a_i(x) f_i(1+\eta_i)}{\sum_{j=0}^m b_j(x) + \sum_{j=0}^m b_j(x)\mu_j} \\
&= \frac{\sum_{i=0}^n a_i(x) f_i(1+\eta_i)}{\sum_{j=0}^m b_j(x)\left(1 + \frac{\sum_{j=0}^m b_j(x)\mu_j}{\sum_{j=0}^m b_j(x)}\right)} \\
&= \frac{\sum_{i=0}^n a_i(x) f_i(1+\eta_i)}{\sum_{j=0}^m b_j(x)}\left(1 - \frac{\sum_{j=0}^m b_j(x)\mu_j}{\sum_{j=0}^m b_j(x)} + O(\epsilon^2)\right) \\
&= \frac{\sum_{i=0}^n a_i(x)\hat{f}_i}{\sum_{j=0}^m b_j(x)} = r(x;\boldsymbol{x},\hat{f}),
\end{aligned}$$

where

$$\hat{f}_i = f_i(1+\eta_i)\left(1 - \frac{\sum_{j=0}^{m} b_j(x)\mu_j}{\sum_{j=0}^{m} b_j(x)} + O(\epsilon^2)\right), \qquad i = 0, \ldots, n.$$

By (23), this means that there exist some $\xi_0, \ldots, \xi_n \in \mathbb{R}$ with $|\xi_0|, \ldots, |\xi_n| \leq (m + B)\max_x \beta(x)\epsilon + O(\epsilon^2)$, such that

$$\hat{f}_i = f_i(1+\eta_i)(1+\xi_i), \qquad i = 0, \ldots, n,$$

and by (14) and (20) we can further assume the existence of some $\varphi_0, \ldots, \varphi_n$ with $|\varphi_0|, \ldots, |\varphi_n| \leq (n + 2 + A)\epsilon + (m + B)\max_x \beta(x)\epsilon + O(\epsilon^2)$, such that

$$\hat{f}_i = f_i(1+\varphi_i), \qquad i = 0, \ldots, n.$$

The statement then follows directly from

$$\begin{aligned}\|\hat{f} - \boldsymbol{f}\|_\infty &= \max_{i=0,\ldots,n} |\hat{f}_i - f_i| = \max_{i=0,\ldots,n} |f_i\varphi_i| \\ &\leq \max_{i=0,\ldots,n} |f_i| \max_{i=0,\ldots,n} |\varphi_i| = \|\boldsymbol{f}\|_\infty \max_{i=0,\ldots,n} |\varphi_i|.\end{aligned}$$

□

The special cases of Theorem 2 for the two different forms of the barycentric rational interpolant then follow with the same reasoning as in Sect. 3.

Corollary 3 *Assume that the weights $w_0, \ldots, w_n$ can be computed as in (24). Then there exists for any $\boldsymbol{f} \in \mathbb{F}^{n+1}$ some $\hat{f} \in \mathbb{F}^{n+1}$ with*

$$\frac{\|\hat{f} - \boldsymbol{f}\|_\infty}{\|\boldsymbol{f}\|_\infty} \leq (n + 4 + W)\epsilon + (n + 2 + W)\max_x \Lambda_n(x; \boldsymbol{x})\epsilon + O(\epsilon^2),$$

for ϵ small enough, such that the second barycentric form in (4) satisfies $\hat{r}(x; \boldsymbol{x}, \boldsymbol{f}) = r(x; \boldsymbol{x}, \hat{f})$.

Corollary 4 *Assume that the weights $w_0, \ldots, w_n$ can be computed as in (24) and the values $\lambda_0(x), \ldots, \lambda_{n-d}(x)$ as in (27). Then there exists for any $\boldsymbol{f} \in \mathbb{F}^{n+1}$ some $\hat{f} \in \mathbb{F}^{n+1}$ with*

$$\frac{\|\hat{f} - \boldsymbol{f}\|_\infty}{\|\boldsymbol{f}\|_\infty} \leq (n + 4 + W)\epsilon + (n - d + C)\max_x \Gamma_d(x; \boldsymbol{x})\epsilon + O(\epsilon^2),$$

for ϵ small enough, such that the first barycentric form in (7) satisfies $\hat{r}(x; \boldsymbol{x}, \boldsymbol{f}) = r(x; \boldsymbol{x}, \hat{f})$.

6 Upper bound for Γ_d

In case of the first barycentric form, the bounds for the forward and backward stability depend on the function Γ_d in (26), and we still need to show that this function is bounded from above. Note that $\Gamma_0 = \Lambda_n$, because $w_i = (-1)^i$ and $\lambda_i = (-1)^i/(x - x_i)$ if $d = 0$, so that the bound for the Lebesgue constant of Berrut's interpolant [8] also holds for Γ_d in this case. In the following, we therefore assume $d \geq 1$ and define

$$N(x) = \sum_{i=0}^{n-d} |\lambda_i(x)| \qquad \text{and} \qquad D(x) = \left| \sum_{i=0}^{n-d} \lambda_i(x) \right|,$$

so that $\Gamma_d(x; \boldsymbol{x}) = N(x)/D(x)$. We further assume that $x \in (x_k, x_{k+1})$ for some $k \in \{0, \ldots, n-1\}$. It then follows from the definition in (8) that all λ_i with index in

$$I_3 = I \cap \{k - d + 1, \ldots, k\},$$

where $I = \{0, \ldots, n-d\}$, have the same sign as $(-1)^{k+d}$ and that the sign alternates for decreasing indices "to the left" and increasing indices "to the right" of I_3. More precisely, the λ_i with index in

$$I_2 = I \cap \{k - d, k - d - 2, \ldots\} \qquad \text{or} \qquad I_4 = I \cap \{k + 1, k + 3, \ldots\}$$

have the same sign as the ones with index in I_3, while the sign is opposite, if the index is in

$$I_1 = I \cap \{k - d - 1, k - d - 3, \ldots\} \qquad \text{or} \qquad I_5 = I \cap \{k + 2, k + 4, \ldots\}.$$

Without loss of generality, we assume that the λ_i are positive for $i \in I_2, I_3, I_4$ and negative for $i \in I_1, I_5$, since multiplying all λ_i with a common constant does not change Γ_d. Letting

$$S_j = \sum_{i \in I_j} \lambda_i(x), \qquad j = 1, \ldots, 5,$$

we then find that

$$N(x) = -S_1 + S_2 + S_3 + S_4 - S_5 \qquad \text{and} \qquad D(x) = S_1 + S_2 + S_3 + S_4 + S_5. \quad (32)$$

To bound Γ_d, we need to bound the sum of the negative λ_i with $i \in I_1, I_5$ as well as the λ_i with $i \in I_3$, which in turn requires to first bound the terms $x - x_i$. To this end, let $h_i = x_{i+1} - x_i$ for $i \in \{0, \ldots, n-1\}$ and define

$$h_{\min} = \min\{h_0, \ldots, h_{n-1}\} \qquad \text{and} \qquad h_{\max} = \max\{h_0, \ldots, h_{n-1}\}.$$

Proposition 1 *For any $i \in \{0, \ldots, n\}$, the distance between $x \in (x_k, x_{k+1})$ and x_i is bounded as*

$$\frac{h_{\min}}{2}(1 + t + 2(k - i)) \leq x - x_i \leq \frac{h_{\max}}{2}(1 + t + 2(k - i)), \qquad i \leq k,$$
$$\frac{h_{\min}}{2}(1 - t + 2(i - k - 1)) \leq x_i - x \leq \frac{h_{\max}}{2}(1 - t + 2(i - k - 1)), \qquad i \geq k + 1,$$

where $t = 2(x - x_k)/h_k - 1 \in (-1, 1)$.

Proof The statement follows directly by noting that

$$x = x_k + \frac{h_k}{2}(1 + t) = x_{k+1} + \frac{h_k}{2}(1 - t)$$

for the given t and because $h_{\min} \leq h_i \leq h_{\max}$ for any $i \in \{0, \ldots, n - 1\}$. □

For bounding the negative λ_i from above, it turns out to be useful to consider them in pairs, with indices from I_1 and I_5 at the same "distance" from I_3.

Lemma 4 *For any $j \in \mathbb{N}$ and $x \in (x_k, x_{k+1})$,*

$$-\lambda_{k-d-2j+1}(x) - \lambda_{k+2j}(x) \leq \left(\frac{1}{h_{\min}}\right)^{d+1}\left(\frac{1}{\prod_{m=0}^{d}(2j + m)} + \frac{1}{\prod_{m=0}^{d}(2j - 1 + m)}\right),$$

where we set $\lambda_i(x) = 0$ for any $i \notin I$.

Proof Since the denominator of $\lambda_{k-d-2j+1}(x)$, for $k - d - 2j + 1 \geq 0$, contains the terms $x - x_{k-2j+1-m}$ for $m = 0, \ldots, d$, it follows from Proposition 1 that

$$\frac{-1}{\lambda_{k-d-2j+1}(x)} \geq \left(\frac{h_{\min}}{2}\right)^{d+1} \prod_{m=0}^{d}(4j + 2m - 1 + t),$$

with $t \in (-1, 1)$ and likewise

$$\frac{-1}{\lambda_{k+2j}(x)} \geq \left(\frac{h_{\min}}{2}\right)^{d+1} \prod_{m=0}^{d}(4j + 2m - 1 - t),$$

for $k + 2j \leq n - d$. Combining both bounds, we get

$$-\lambda_{k-d-2j+1}(x) - \lambda_{k+2j}(x) \leq \left(\frac{2}{h_{\min}}\right)^{d+1} g(t),$$

where

$$g(t) = \frac{1}{\prod_{m=0}^{d}(4j + 2m - 1 + t)} + \frac{1}{\prod_{m=0}^{d}(4j + 2m - 1 - t)}.$$

As g is clearly even and

$$g(1) = \frac{1}{2^{d+1}}\left(\frac{1}{\prod_{m=0}^{d}(2j+m)} + \frac{1}{\prod_{m=0}^{d}(2j-1+m)}\right),$$

it remains to show that $g(t) \leq g(1)$ for $t \in [0,1]$. To this end, note that $g(t) = p(t)/q(t)$ for

$$p(t) = \prod_{m=0}^{d}(4j+2m-1+t) + \prod_{m=0}^{d}(4j+2m-1-t)$$

and

$$q(t) = \prod_{m=0}^{d}\left((4j+2m-1)^2 - t^2\right).$$

By the product rule,

$$p'(t) = \sum_{l=0}^{d}\left(\prod_{m=0,\, m\neq l}^{d}(4j+2m-1+t) - \prod_{m=0,\, m\neq l}^{d}(4j+2m-1-t)\right)$$

and

$$q'(t) = -2t\sum_{l=0}^{d}\prod_{m=0,\, m\neq l}^{d}\left((4j+2m-1)^2 - t^2\right).$$

For $t \in [0,1]$, we observe that $p(t) > 0$, $q(t) > 0$, $p'(t) \geq 0$, and $q'(t) \leq 0$, hence

$$p'(t)q(t) \geq 0 \geq p(t)q'(t),$$

and it follows from the quotient rule that g is monotonically increasing over $[0,1]$. □

Next, let us bound the λ_i with indices in I_3 from below.

Lemma 5 *For any $i \in I_3$ and $x \in (x_k, x_{k+1})$,*

$$\lambda_i(x) \geq \left(\frac{1}{h_{\max}}\right)^{d+1}\frac{4}{d!}. \tag{33}$$

Proof Since the denominator of $\lambda_i(x)$, for $i \in I_3$, contains the factors $x - x_{k-m}$ for $m = 0, \ldots, k-i$ and the factors $x_{k+1+l} - x$ for $l = 0, \ldots, i+d-k-1$, we conclude from Proposition 1 that

$$\begin{aligned}\frac{1}{\lambda_i(x)} &\le \left(\frac{h_{\max}}{2}\right)^{d+1} \prod_{m=0}^{k-i}(1+t+2m) \prod_{l=0}^{i+d-k-1}(1-t+2l)\\ &= \left(\frac{h_{\max}}{2}\right)^{d+1} (1-t^2) \prod_{m=1}^{k-i}(1+t+2m) \prod_{l=1}^{i+d-k-1}(1-t+2l)\\ &\le \left(\frac{h_{\max}}{2}\right)^{d+1} \prod_{m=1}^{k-i}(2+2m) \prod_{l=1}^{i+d-k-1}(2+2l)\\ &= \left(\frac{h_{\max}}{2}\right)^{d+1} 2^{d-1}(k-i+1)!(i+d-k)!,\end{aligned}$$

and the statement then follows, because $a!b! \le (a+b-1)!$ for any $a, b \in \mathbb{N}$. □

We are now ready to derive a general upper bound on the function Γ_d in (26), which turns out to depend on d and the *mesh ratio*

$$\mu = \frac{h_{\max}}{h_{\min}}.$$

Theorem 3 *If $d \ge 1$, then*

$$\Gamma_d(x; \boldsymbol{x}) \le 1 + \frac{\mu^{d+1}}{2d} \tag{34}$$

for any set of ascending interpolation nodes $\boldsymbol{x} = (x_0, \ldots, x_n) \in \mathbb{R}^{n+1}$ and any $x \in [x_0, x_n]$.

Proof If $x = x_k$ for some $k \in \{0, \ldots, n\}$, then, after multiplying both $N(x)$ and $D(x)$ by $\prod_{i=0}^{n} |x - x_i|$, we find that $\Gamma_d(x; \boldsymbol{x}) = 1$, which is clearly smaller than the upper bound in (34). Otherwise, there exists some $k \in \{0, \ldots, n-1\}$, such that $x \in (x_k, x_{k+1})$, and it follows from Lemma 4 that

$$\begin{aligned}-S_1 - S_5 &\le \sum_{j=1}^{\infty} \left(\frac{1}{h_{\min}}\right)^{d+1} \left(\frac{1}{\prod_{m=0}^{d}(2j+m)} + \frac{1}{\prod_{m=0}^{d}(2j-1+m)}\right)\\ &= \left(\frac{1}{h_{\min}}\right)^{d+1} \sum_{j=1}^{\infty} \prod_{m=0}^{d} \frac{1}{j+m}\\ &= \left(\frac{1}{h_{\min}}\right)^{d+1} \frac{1}{d \cdot d!},\end{aligned}$$

where the sum of the series can be found in [17, p. 464]. Together with Lemma 5, this implies

$$\frac{-2(S_1 + S_5)}{\lambda_i(x)} \le \frac{\mu^{d+1}}{2d}$$

for any $i \in I_3$. As $\lambda_i(x) \le S_3 \le D(x)$ and $N(x) - D(x) = -2(S_1 + S_5)$, we conclude that

$$N(x) - D(x) \le \frac{\mu^{d+1}}{2d} D(x),$$

and the statement then follows immediately. □

In the case of equidistant nodes, when the mesh ratio is $\mu = 1$, the upper bound in (34) is simply $1 + 1/(2d)$, so it becomes smaller as d grows. For other nodes, μ may depend on n, which may result in very large upper bounds. For example, in the case of Chebyshev nodes, one can show that μ grows asymptotically linear in n. However, our numerical experiments suggest that the function Γ_d is always small, and we believe that the upper bound in Theorem 3 can be improved significantly in future work.

7 Numerical experiments

We performed numerous experiments to verify the results proven in the previous sections numerically and report some representative results below. In particular, we analyze the various algorithms that implement the first barycentric form (7) both in terms of stability and computational cost (Sect. 7.1) and focus on the comparison between the first and the second form for an example where $\Lambda_n \gg \Gamma_d$ (Sect. 7.2).

Our experimental platform is a Windows 10 laptop with 1.8 GHz Intel Core i7-10510U processor and 16 GB RAM, and we implemented all algorithms in *C++*. In what follows, the 'exact' values were computed in multiple-precision (1024 bit) floating point arithmetic using the *MPFR* library [18], while all other values were computed in standard double precision. Moreover, we took care of providing all input data (interpolation nodes, data values, and evaluation points) in double precision, so that they do not cause any additional error.

7.1 Comparison of algorithms for the first barycentric form

For the first example, we consider the case of $n+1$ interpolation nodes $x_i = \mathrm{fl}(y_i) \in [-1, 1]$ for $i = 0, \ldots, n$, derived from the equidistant nodes $y_i = 2i/n - 1$ by rounding them to double precision, and associated (rounded) data $f_i = \mathrm{fl}(f(y_i))$, sampled from the test function

$$f(x) = \tfrac{3}{4} e^{-\frac{(9x-2)^2}{4}} + \tfrac{3}{4} e^{-\frac{(9x+1)^2}{49}} + \tfrac{1}{2} e^{-\frac{(9x-7)^2}{4}} + \tfrac{1}{5} e^{-(9x-4)^2},$$

and we compare three ways of evaluating the first barycentric form in (7), which differ in the way the denominator is computed.

The first algorithm simply evaluates the functions λ_i as in (8), leading to the error mentioned in Lemma 2 and then sums up these values to get the denominator. The second algorithm by Camargo [11, Section 4.1] instead increases the stability of the

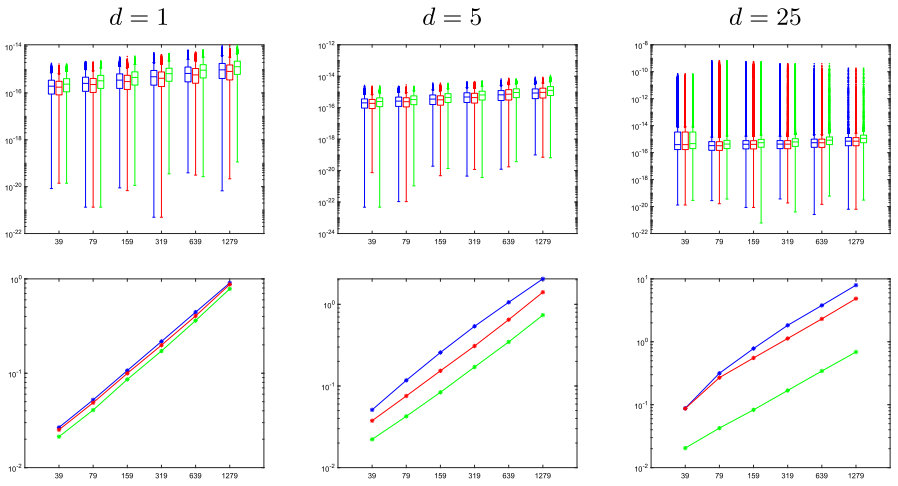

Fig. 1 Distribution of the relative forward errors of the first barycentric form for equidistant nodes at 50,000 random evaluation points (top) and overall running time in seconds (bottom), both on a logarithmic scale, for different n and three choices of d (left, middle, right), using the standard algorithm (blue), Camargo's algorithm (red), and our efficient variant of the standard algorithm (green) (color figure online)

summation by first computing sums of pairs of λ_i's such that all these sums have the same sign. The third algorithm implements our iterative strategy in (31) before taking the sum, which is more efficient than the first algorithm, but less precise (cf. Lemma 3). All three algorithms compute the numerator of the first barycentric form in the same way, first dividing the weights w_i by $x - x_i$, then multiplying the results by f_i, and finally summing up these values. The w_i themselves are precomputed with the pyramid algorithm in (29) and (30). Note that, although the weights for equidistant nodes are integer multiples of each other in theory [6], they do not have this property in this example, because the nodes x_i are not exactly equidistant, because of the rounding.

To compare these three algorithms, we used them to evaluate the barycentric rational interpolant with weights in (6) for $d \in \{1, 5, 25\}$ and an increasing number of interpolation nodes, $n \in \{39, 79, 159, 319, 639, 1279\}$, at 50,000 random points from $[-1+10^3\epsilon, 1-10^3\epsilon] \setminus \{x_0, \ldots, x_n\}$. Figure 1 shows the corresponding running times and the distribution of the relative forward error of the computed values. For the latter, we chose a box plot, where the bottom and top of each box represent the interquartile range, from the first to the third quartile, and the line in the middle shows the median of the relative forward errors. The whiskers range from the minimum to the maximum, excluding those values that are more than 1.5 times the interquartile range greater than the third quartile, which are instead considered outliers and shown as isolated points.

On the one hand, we observe that Camargo's algorithm beats the standard algorithm in terms of running time, but that our efficient algorithm is the fastest, especially as d grows, because its time complexity does not depend on d. On the other hand, our efficient algorithm is less precise than the standard algorithm, as predicted by Lemma 3 and Camargo's algorithm gives the smallest errors, except for $d = 5$ and $n \in \{639, 1279\}$. Nevertheless, the computations confirm the forward stability for all

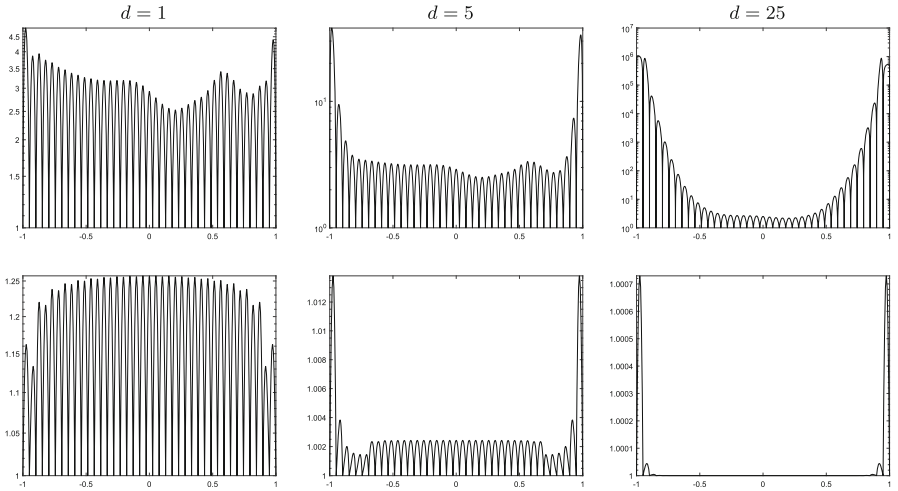

Fig. 2 Plots of $\kappa(x; \boldsymbol{x}, \boldsymbol{f})$ (top) and $\Gamma_d(x; \boldsymbol{x})$ (bottom) for equidistant nodes and $x \in [-1, 1]$, both on a logarithmic scale, for $n = 39$ and three choices of d (left, middle, right)

three algorithms. For Camargo's algorithm, this follows from the backward stability, which is proven in [11], and for the other two algorithms it is implied by Corollary 2 and Lemmas 1–3.

The rather large errors of the outliers in the case $d = 25$ can be explained by the behaviour of the componentwise relative condition number κ, shown in Figure 2. While κ is small for all $x \in [-1, 1]$ in the case of $d = 1$, it starts to grow considerably close to the end points of the interval as d grows, up to 10^6 for $n = 39$ and 10^8 for $n = 1279$ in the case of $d = 25$, and so does the upper bound on the relative forward error in (28). While this upper bound also depends on Γ_d, Figure 2 shows that this function is always small and its maximum even decreases as d grows, independently of n. This is in agreement with Theorem 3, because $\mu \approx 1$ in this example. The fact that the maximum error still seems to decrease for $d = 25$ as n increases is simply due to the fact that the 50,000 sample points are not sufficiently many to "catch" the worst case, because the region near the endpoints where κ grows rapidly actually shrinks as n grows.

Of course, it is also possible to evaluate the rational interpolant using the second barycentric form (4), which is actually the best choice for this example, giving relative forward errors that are similar to the ones of the standard algorithm for the first barycentric form, but being roughly twice as fast as the efficient algorithm, both of which is not surprising. Regarding the efficiency, the second form is superior, because the denominator can be computed "on-the-fly" at almost no extra cost during the evaluation of the numerator. As for the error, we note that Λ_n is much smaller than κ for the nodes used in this example [10] and so the upper bound in (25) is dominated by κ, exactly as the upper bound in (28). However, for non-uniform nodes, the situation can be quite different, as the next example shows.

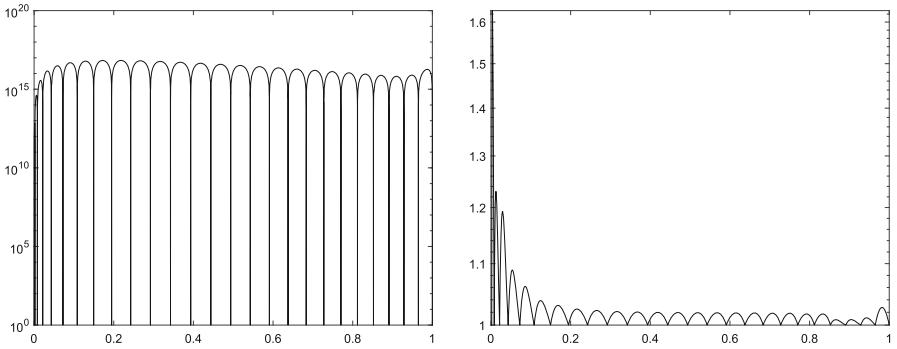

Fig. 3 Plots of $\Lambda_n(x; \boldsymbol{x})$ (left) and $\Gamma_d(x; \boldsymbol{x})$ (right) for a non-regular distribution of nodes and $x \in [0, 1]$, both on a logarithmic scale

7.2 Worst-case comparison of first and second barycentric form

The aim of the second example is to compare the standard algorithm for the first barycentric form in (7), as described in Sect. 7.1, with a straightforward implementation of the second barycentric form, following the formula in (4). The weights w_i are again precomputed with the pyramid algorithm.

We consider $n = 29$, $d = 3$, and interpolation nodes $x_i = \mathrm{fl}(y_i) \in [0, 1]$ for $i = 0, \ldots, n$, obtained by rounding to double precision the values $y_i = F(t_i)$, where

$$F(t) = \begin{cases} 0, & t = 0, \\ e^{1-\frac{1}{t}}, & t \in (0, 1] \end{cases}$$

and $t_i = i/n$. We choose these nodes, because the functions Λ_n and Γ_d behave completely differently in this case, as shown in Fig. 3. While Λ_n reaches huge values, up to 10^{17}, Γ_d is small (even though the latter is not guaranteed by Theorem 3). Hence, the upper bounds in Corollaries 1 and 2 suggest that we may see a big difference in the forward stability of the first and the second barycentric form, if we choose the data such that the condition number κ is small.

One such choice, which is also presented by Higham for the case $d = n$ with equidistant nodes [1], is to take the data

$$f_i = \begin{cases} 0, & i = 0, \ldots, n-1, \\ 1, & i = n, \end{cases}$$

which can be interpreted as having been sampled from the nth Lagrange basis polynomial $l_n(x) = \prod_{i=0}^{n-1} \frac{x-x_i}{x_n-x_i}$, that is, $f_i = \mathrm{fl}(l_n(y_i))$. For this data, we know that $\kappa(x; \boldsymbol{x}, \boldsymbol{f}) = 1$ for all $x \in [0, 1]$, so the upper bounds on the relative forward errors in (25) and (28) are dominated by Λ_n and Γ_d, respectively. Consequently, as shown in Fig. 4, the barycentric rational interpolant is reproduced faithfully by the first form, but not by the second, because the relative forward error for the first form is on the order of ϵ, while it can be on the order of 1 for the second form; see Fig. 5 (left).

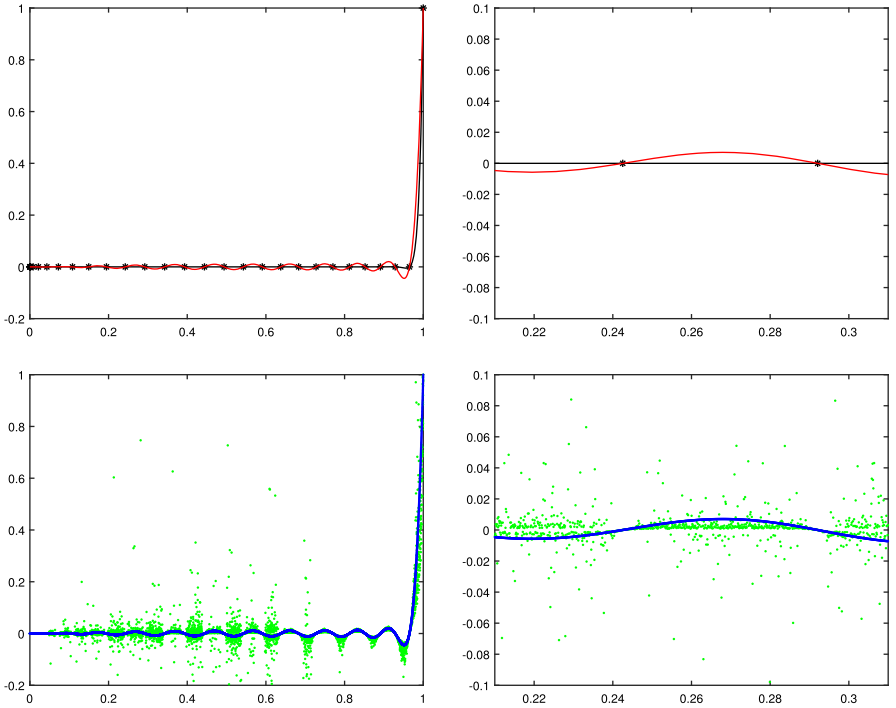

Fig. 4 Plots of $l_n(x)$ (black) and the barycentric rational interpolant $r(x)$ for $d = 3$ (red) for non-regularly distributed interpolation nodes over the whole interval [0, 1] (top left) and a close-up view over [0.21, 0.31] (top right). Evaluating $r(x)$ at 10,000 equidistant evaluation points in $[10^3\epsilon, 1 - 10^3\epsilon]$ with the standard implementations of the first (blue dots) and the second (green dots) barycentric form shows that the first form is stable, while the second form is not (bottom) (color figure online)

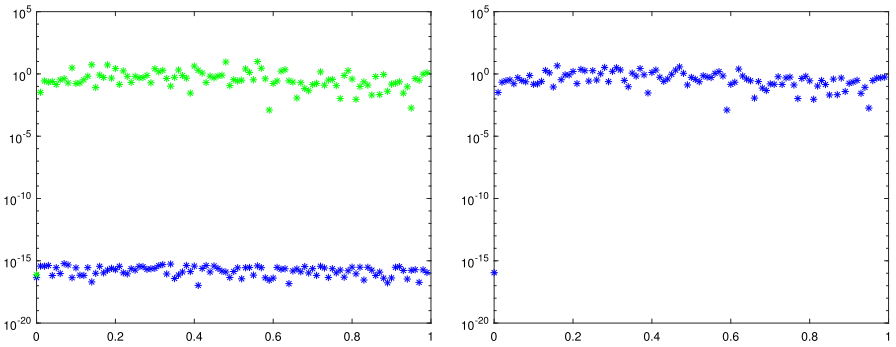

Fig. 5 Plot of relative forward errors of the first (blue) and second (green) barycentric form for a non-regular distribution of nodes at 100 equidistant evaluation points in $[10^3\epsilon, 1-10^3\epsilon]$ with data sampled from the nth Lagrange basis polynomial (left) and the constant one function (right). Since both plots are on a logarithmic scale and the second form is exact in the latter case, the corresponding errors are missing in the plot on the right (color figure online)

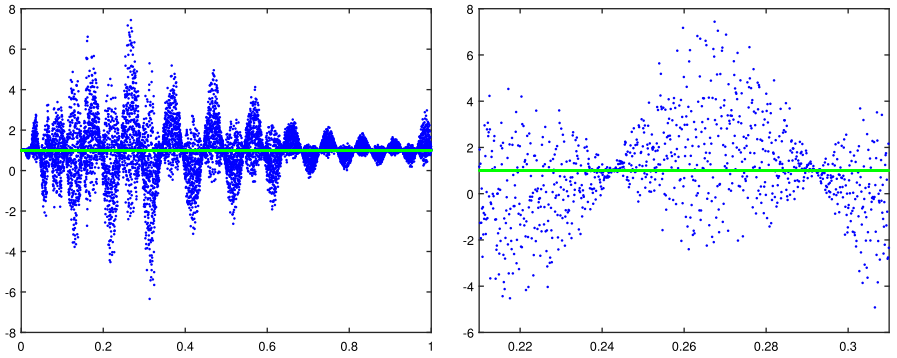

Fig. 6 Even though the barycentric rational interpolant of the constant one function for non-regularly distributed interpolation nodes is simply $r(x) = 1$, evaluating it at 10, 000 equidistant evaluation points in $[10^3\epsilon, 1 - 10^3\epsilon]$ shows that the second form (green dots) is stable, while the first form (blue dots) is not

However, the opposite may happen as well. If we consider the data

$$f_i = 1, \qquad i = 0, \ldots, n,$$

sampled from the constant one function, then $\kappa = \Lambda_n$, and the upper bounds in (25) and (28) suggest that both forms can be unstable, even though the barycentric rational interpolant is simply $r(x) = 1$. Figure 5 (right) and Figure 6 confirm that this is indeed the case for the first barycentric form. However, the second barycentric form is perfectly stable, because the numerator and the denominator in (4) are identical and cancel out to always give the correct function value 1.

8 Conclusion

Barycentric interpolation offers a fast and stable means of evaluating the polynomial Lagrange interpolant using either the first or the second barycentric form. While the first form is always numerically stable, the second form is stable only for interpolation nodes with a small Lebesgue constant.

Evaluating a rational interpolant via the second barycentric form comes with the same limitation, but for the special family of barycentric rational interpolants with weights in (6), a computationally more attractive first barycentric form is available. Instead of depending on the Lebesgue constant, both the forward and the backward stability of a straightforward implementation of this first barycentric form depend on the function Γ_d in (26).

Unlike the Lebesgue constant, Theorem 3 shows that the maximum of Γ_d is independent of n. Moreover, it is guaranteed to be very small for equidistant nodes, regardless of d, while the Lebesgue constant is known to grow logarithmically in n and exponentially in d in this case. Based on our numerical experiments, the maximum of Γ_d seems to be small for other distributions of interpolation nodes, too, even if the mesh ratio μ is big, as in the example in Sect. 7.2, and we believe that the upper bound

in (34) can be improved considerably in future work. For example, if $d = n$, then Γ_d is just the constant one function, independent of μ.

Overall, we conclude that the most efficient way of stably evaluating a barycentric rational interpolant with weights in (6) is by determining the weights with the pyramid algorithm in (29) and (30) in a preprocessing step, and, for every evaluation point x, first computing the values $\lambda_i(x)$ with our iterative strategy in (31) and then the value of the interpolant using a straightforward implementation of the first barycentric form in (7). Alternatively computing the sum of the $\lambda_i(x)$ in the denominator with Camargo's algorithm [11] results in slightly smaller forward errors, but is significantly slower, especially for larger d.

Acknowledgements This work was supported by the Swiss National Science Foundation (SNSF) under project number.

Funding Open access funding provided by Università della Svizzera italiana

References

1. Higham, N.J.: The numerical stability of barycentric Lagrange interpolation. IMA J. Numer. Anal. **24**(4), 547–556 (2004). https://doi.org/10.1093/imanum/24.4.547
2. Berrut, J.-P., Trefethen, L.N.: Barycentric Lagrange interpolation. SIAM Rev. **46**(3), 501–517 (2004). https://doi.org/10.1137/S0036144502417715
3. Berrut, J.-P., Mittelmann, H.D.: Lebesgue constant minimizing linear rational interpolation of continuous functions over the interval. Comput. Math. Appl. **33**(6), 77–86 (1997). https://doi.org/10.1016/S0898-1221(97)00034-5
4. Schneider, C., Werner, W.: Some new aspects of rational interpolation. Math. Comput. **47**(175), 285–299 (1986). https://doi.org/10.1090/S0025-5718-1986-0842136-8
5. Berrut, J.-P.: Rational functions for guaranteed and experimentally well-conditioned global interpolation. Comput. Math. Appl. **15**(1), 1–16 (1988). https://doi.org/10.1016/0898-1221(88)90067-3
6. Floater, M.S., Hormann, K.: Barycentric rational interpolation with no poles and high rates of approximation. Numer. Math. **107**(2), 315–331 (2007). https://doi.org/10.1007/s00211-007-0093-y
7. Salazar Celis, O.: Practical rational interpolation of exact and inexact data: theory and algorithms. PhD thesis, Department of Computer Science, University of Antwerp (2008)
8. Bos, L., De Marchi, S., Hormann, K., Sidon, J.: Bounding the Lebesgue constant for Berrut's rational interpolant at general nodes. J. Approx. Theory **169**, 7–22 (2013). https://doi.org/10.1016/j.jat.2013.01.004
9. Bos, L., De Marchi, S., Hormann, K., Klein, G.: On the Lebesgue constant of barycentric rational interpolation at equidistant nodes. Numer. Math. **121**(3), 461–471 (2012). https://doi.org/10.1007/s00211-011-0442-8
10. Hormann, K., Klein, G., De Marchi, S.: Barycentric rational interpolation at quasi-equidistant nodes. Dolomit. Res. Notes Approx. **5**(1), 1–6 (2012). https://doi.org/10.14658/pupj-drna-2012-1-1 https://doi.org/10.14658/pupj-drna-2012-1-1
11. de Camargo, A.P.: On the numerical stability of Floater–Hormann's rational interpolant. Numer. Algorithms **72**(1), 131–152 (2016). https://doi.org/10.1007/s11075-015-0037-z

12. Hormann, K., Schaefer, S.: Pyramid algorithms for barycentric rational interpolation. Comput. Aided Geom. Des. **42**, 1–6 (2016). https://doi.org/10.1016/j.cagd.2015.12.004
13. Mascarenhas, W., Camargo, A.: The backward stability of the second barycentric formula for interpolation. Dolomit. Res. Notes Approx. **7**(1), 1–12 (2014). https://doi.org/10.14658/pupj-drna-2014-1-1
14. Trefethen, L.N., Bau, D.: Numerical Linear Algebra. SIAM, Philadelphia (1997)
15. Gohberg, I., Koltracht, I.: Mixed, componentwise, and structured condition numbers. SIAM J. Matrix Anal. Appl. **14**(3), 688–704 (1993). https://doi.org/10.1137/0614049
16. Higham, N.J.: Accuracy and Stability of Numerical Algorithms, 2nd edn. SIAM, Philadelphia (2002)
17. Bronshtein, I.N., Semendyayev, K.A., Musiol, G., Mühlig, H.: Handbook of Mathematics, 6th edn. Springer, Berlin (2015)
18. Fousse, L., Hanrot, G., Lefèvre, V., Pélissier, P., Zimmermann, P.: MPFR: A multiple-precision binary floating-point library with correct rounding. ACM Trans. Math. Softw. (2007). https://doi.org/10.1145/1236463.1236468