Maple Transactions

Blends have decent numerical properties

ROBERT M. CORLESS, University of Western Ontario, Canada

A "blend" is a two-point Hermite interpolational polynomial, typically of quite high degree. This note shows that implementing them in a double Horner evaluation scheme has good backward error, and also shows that the Lebesgue constant for a balanced blend or nearly balanced blend on the interval $[0, 1]$ is bounded by 2, independently of the grade or degree of the approximation. On $[-1, 1]$, which is a more natural interval for comparison, it is of course unbounded, but grows only like $2\sqrt{m/\pi}$ where $2m + 1$ is the grade of approximation. I also show that the quadrature schemes for balanced blends amplify errors only by $O(\ln m)$.

CCS Concepts: • **Computing methodologies** → **Representation of mathematical objects**; **Representation of mathematical functions**.

Additional Key Words and Phrases: Blends, Hermite interpolational polynomials, backward error, Lebesgue constant, integration

Recommended Reference Format:
Robert M. Corless. 2023. Blends have decent numerical properties. *Maple Trans.* X, Y, Article Z (2023), 20 pages.
https://doi.org/10.5206/mt.vXiY.Z

1 Introduction

A blend is a two-point Hermite interpolant, typically of quite high degree [6]. Blends can be used by themselves to approximate functions whose Taylor coefficients are known at *two* points, or they can be used pairwise along a polygonal path in the complex plane between "knots" at which Taylor coefficients are known. In that latter case I refer to the approximant as a "blendstring"; if there are only two knots I just use the word "blend," because "Two-point Hermite interpolational polynomial" is a mouthful.

The idea is very old, going back to Hermite's Cours d'Analyse, and indeed one important formula appears already on page 4 of that venerable volume [10]. Nonetheless I think it has been somewhat unjustly neglected[1]. Blends have some quite interesting approximation theoretic properties, and they are quite convenient to work with computationally. I believe that the theorems in this present paper are new, although of course, given the wide spread of more than a century of literature in many languages, it is difficult to be sure. I do *not* think that the integration formula in section 5 can possibly be new, although I have not seen it in the literature. It seems like the kind of thing that Hermite would have known.

In more recent history, Erik Postma and I introduced a Maple program to efficiently evaluate, differentiate, and integrate blends in [6]. Chris Brimacome, Mair Zamir and I used them to compute Mathieu functions [2], and are currently using those to solve a problem in hemodynamics [3]. Other tools could have been used, but blends and blendstrings allow very high-precision computations and controllable and verifiable accuracy, which was, and is, important to us.

[1] I've been thinking about the reasons for this. Perhaps the main one is that blends do not simplify (or even make much sense) in the limit as the grade (degree) goes to infinity. The formula is also, admittedly, a bit more complicated.

Author's address: Robert M. Corless, rcorless@uwo.ca, University of Western Ontario, London, Ontario, Canada, N6A 5B9.

Special cases of blends, such as the smoothstep function, and cubic and quintic blends, are apparently useful in graphics and in Computer-Aided Geometric Design. I believe that blends and blendstrings may have other uses in approximation, chiefly when a high degree of smoothness is present and a high degree of accuracy is needed.

2 Definitions and basic properties

Much of this section is based on [6]; an open version of that paper is available at https://arxiv.org/abs/2007.05041, if the reader wishes more detail. As is common in the literature of matrix polynomial eigenvalue problems, in what follows I use the word *grade* to mean "degree at most". That is, a polynomial of grade (say) 5 is of degree at most 5. But because here the leading coefficient is not visible, we don't know the exact degree, which could be lower.

Consider an analytic function $f(z)$ with Taylor series coefficients known at $z = a$ and at $z = b$. Convert to the unit interval by introducing a new variable s with $z = a + s(b - a)$. This conversion has some interesting consequences for approximation, as we will see.

2.1 The basic formula

The following formula, known already to Hermite [10, p. 4], has the property that the grade $m+n+1$ polynomial

$$H_{m,n}(s) = \sum_{j=0}^{m} \left[\sum_{k=0}^{m-j} \binom{n+k}{k} s^{k+j} (1-s)^{n+1} \right] p_j + \sum_{j=0}^{n} \left[\sum_{k=0}^{n-j} \binom{m+k}{k} s^{m+1} (1-s)^{k+j} \right] (-1)^j q_j \tag{1}$$

has a Taylor series matching the given $m+1$ values $p_j = f^{(j)}(0)/j!$ at $s = 0$ and another Taylor series matching the given $n+1$ values $q_j = f^{(j)}(1)/j!$ at $s = 1$. Putting this in symbolic terms and using a superscript (j) to mean the jth derivative with respect to s gives

$$\frac{H^{(j)}_{m,n}(0)}{j!} = p_j \qquad \text{and} \qquad \frac{H^{(j)}_{m,n}(1)}{j!} = q_j$$

for $0 \le j \le m$ on the left and for $0 \le j \le n$ on the right. This is a kind of interpolation, indeed a special case of what is called *Hermite* interpolation. As with Lagrange interpolation, where for instance two points give a grade one polynomial, that is, a line, here $m+n+2$ pieces of information gives a grade $m+n+1$ polynomial. As per [6] this formula can be evaluated in $O(m+n)$ arithmetic operations, and this paper will assume that the Taylor coefficients are known to a fixed working precision, most commonly 15 decimal digits. If one wants to work in higher precision, it will be necessary to start with Taylor coefficients evaluated to that higher precision.

2.2 Example

Consider "Robin Crusoe's" example[2] from [5, Ch. 3], namely approximation of

$$f(u) = \ln\left(\frac{1+2u/3}{1-2u/3}\right) \tag{2}$$

over the interval $-1 \le u \le 1$. This function is *even* and so we may reduce the interval to $0 \le u \le 1$. The treatment in [5] settled on approximation by Chebyshev polynomials using Lanczos' τ-method,

[2] I wrote a long shaggy-dog story in that chapter about approximating this function using only scratches in sand and lots of time. I had fun, and I worked hard to make sure that no pronoun whatever was used; sadly nobody seems to have noticed. Oh, well!

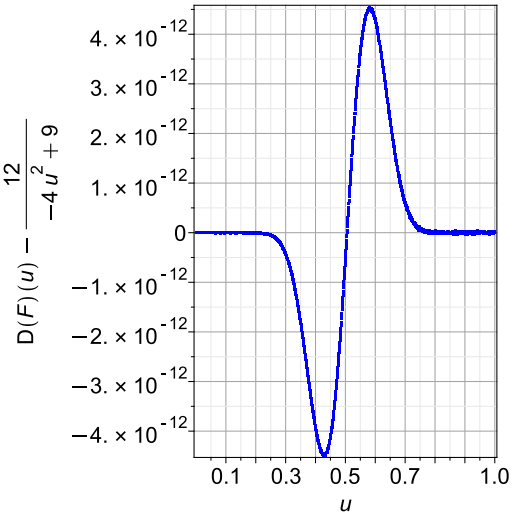


Fig. 1. The error $y' - f'(u)$ in the derivative of the $(21, 21)$ blend approximating $f(u) = \ln((1 - 2u/3)/(1 + 2u/3))$ on $0 \le u \le 1$. The derivative was taken "semi-automatically" by differentiating the numerically stable loops used to evaluate $f(u)$.

but a blend works pretty well, as we see here. Using a balanced blend of grade 21 on each end of the interval $0 \le u \le 1$, I evaluated the blend and its first derivative at 2023 equally-spaced points. In Figure 1 I plot the error in the computed derivative. The line is a bit blurry on the u-axis at the right, showing effects of rounding error just beginning. Nonetheless the approximation is quite satisfactory, especially because it used only geometric series at either end.

3 Approximation theoretic properties

Recall that the Lebesgue function $L(s)$ for a polynomial basis $\phi_j(s)$ for $0 \le j \le N$ is the function $L(s) = \sum_{j=0}^{N} |\phi_j(s)|$. Relative errors δ_j in polynomial coefficients $a_j(1 + \delta_j)$ produce changes $\Delta p(s)$ in the polynomial value. By the triangle inequality, $|\Delta p(s)| \le L(s)\|\mathbf{a}\|_\infty \varepsilon$ if all relative coefficient changes $|\delta_j| \le \varepsilon$. See for instance [12]. It was claimed in [6] that for a *balanced* blend, that is one where the grade m of Taylor polynomial at the left end is the same as the grade n at the right end, $L(s) \le 2$ on $0 \le s \le 1$, independently of the grade m.

This is a somewhat remarkable statement. The reader may recall that many polynomial bases have Lebesgue constants that are exponential in the degree. The Chebyshev basis, like several other excellent bases for interpolation, have Lebesgue constants that grow only logarithmically like the degree: $\Lambda_n = 2\ln(n)/\pi + O(1)$. Bernstein's theorem [15, Chapter 15] says that the Lebesge constant must grow at least logarithmically with the degree.

Here we have a constant. This seems like a contradiction. It's not, and 2 is not even the best constant: the Bernstein basis has Lebesgue constant 1 on this interval. But 2 is still good enough to be interesting.

But of course, this is sleight-of-hand: we are working on the interval $[0, 1]$ whereas Bernstein's theorem works on $[-1, 1]$, which is a more natural interval in many ways: it is the projection of the complex unit circle to the real line. If we translate our results to the interval $[-1, 1]$, we pick up factors of 2 in annoying ways. This is because if $x = -1 + 2s$ with $s \in [0, 1]$ so $x \in [-1, 1]$, then

$d/ds = 2d/dx$. Unit Taylor coefficients in the x variable become vectors of powers of 2 in the s variable. So this result cannot be as good as it seems at first. We will see that it's not so bad, though.

We also claimed in [6] that a double Horner evaluation was backward stable on $0 \le s \le 1$ in the following sense: namely, that numerical evaluation with unit roundoff u gave the *exact* answer for a blend with coefficients changed at most by a modest multiple of u. We did not give the proof there, but I will here. This claimed theorem, together with the claimed bound on $L(s)$, guarantees accurate approximation using IEEE floating-point arithmetic.

This paper gives details for the proofs of each claim.

REMARK 3.1. *Analysis of the impact of errors in the data (coefficients) by using Lebesgue functions and constants as we do here is a bit weaker than is possible. In* [8]*, a sharper theory is introduced which takes into account how the coefficients change as the basis is changed, and under these sharper circumstances they prove that among all bases which are nonnegative on the interval in question (often* $[0,1]$*), the Bernstein bases are optimal; and if any other basis has as good errors, then it must be equivalent to the Bernstein basis. In* [7] *this theory was extended to Lagrange bases, which are nonnegative on a set of nodes in an interval, and these can be "even better." This result was rediscovered in* [4] *and further analyzed. Because the basis for a blend is nonnegative on* $[0,1]$ *the Farouki–Goodman theory shows that blends cannot have as good numerical properties as Bernstein bases do. Nonetheless we will see here that they are not so bad.*

3.1 Truncation error

The truncation error in approximating $f(z)$ by the blend is given by (in the scaled variable s)

$$f(s) - H_{m,n}(s) = \frac{f^{(m+n+2)}(\theta)}{(m+n+2)!} s^{m+1}(s-1)^{n+1} \,. \tag{3}$$

Here $\theta \in (0,1)$ is otherwise unknown. A complex valued version of this error formula (due to Hermite) is available:

$$f(s) - H_{m,n}(s) = \frac{1}{2\pi i} \oint \frac{s^{m+1}(1-s)^{n+1} f(\zeta)}{\zeta^{m+1}(1-\zeta)^{n+1}(\zeta - s)} \, d\zeta \,, \tag{4}$$

where the counterclockwise integration takes place on a closed contour that encloses s, 0, and 1, and no singularities of f. The quality of the approximation depends very strongly on how big a contour one can make. The ratio of the node polynomials $(s/\zeta)^{m+1}((1-s)/(1-\zeta))^{n+1}$ will be exponentially small if the contour, where ζ lives, is a long way from where s is (typically exactly on the segment $[0,1]$). Any nearby singularities of f will restrict that. Blends are best for entire functions, really. As discussed in [15, Chapter 11], they are not as accurate as Chebyshev polynomial interpolation by a factor of 2^{m+n}; still, they are better than plain Taylor approximation, by a factor of 2^{m+n}. In some sense, then, they are mid-way between the two, in terms of accuracy.

The maximum value of the polynomial $s^{m+1}(1-s)^{n+1}$ on the interval $(0,1)$ is attained at the point $s = (m+1)/(m+n+1)$ and is $(m+1)^{m+1}(n+1)^{n+1}/(m+n+2)^{m+n+2}$, which if $m = n$ reduces to $2^{-2(m+1)}$. One frequently uses $m = 10$ or more, and so this factor reduces the error by a factor of a million or more, compared to a Taylor series expansion of grade $m+n+2$ on only one side. In some sense this factor is the real reason blends are interesting.

3.2 Evaluation and differentiation of a blend

The paper [6] implemented a double Horner expansion of the formula (1). The algorithm for the half-sum function HSF is recorded in Algorithm 1, so that $H_{m,n}(s) = \mathrm{HSF}(s, m, n, p_j) + \mathrm{HSF}(1-s, n, m, q_j(-1)^j)$. We will see a proof here that execution of this algorithm is numerically stable in that it gives the *exact* evaluation of a blend of slightly perturbed Taylor coefficients $p_j(1+\delta_{p,j})$ and

$q_j(1+\delta_{q,j})$. We will see explicit bounds on the perturbations in terms of modest multiples of the unit roundoff u.

Several examples were given in [6] showing the method's numerical stability in practice. As stated earlier, the cost of evaluation is linear in the grade of the blend, $O(m+n)$. Since the Horner-like evaluation is simply a pair of for-loops, automatic differentiation is straightforward, and the user can request derivatives of the blend to be delivered with the values. Experience shows that rounding errors in the automatic derivatives are similarly small, even though differentiation is infinitely ill-conditioned.

3.3 Notation for rounding error analysis

I use the notation described in [11, Chapter 3]. For convenience, I define the quantities here. I also assume IEEE 854 arithmetic[3], which satisfies the model for "op" being any of +, −, ·, or /,

$$\mathrm{fl}(a \text{ op } b) = (a \text{ op } b)(1+\delta) \tag{5}$$

with $|\delta| \leq u$ where u is the *unit roundoff*. For a typical double precision IEEE *754* computation, $u = 2^{-53} \approx 10^{-16}$. For Maple, setting the environment variable **`Digits`** larger than 15 effectively sets $u = 5 \cdot 10^{-1-\mathrm{Digits}}$. For single precision, $u = 2^{-24} \approx 6 \cdot 10^{-8}$ and for half precision $u = 2^{-11} \approx 5 \cdot 10^{-4}$. The notation $\mathrm{fl}(X)$ means the floating-point result of operation X.

The notation of Lemma 3.1 of [11] is quite convenient. I reproduce that Lemma here:

Lemma 1 (3.1 of of [11]). *If $|\delta_i| \leq u$ and $\rho_i = \pm 1$ for $1 \leq i \leq n$, and $nu < 1$,*

$$\prod_{i=1}^{n}(1+\delta_i)^{\rho_i} = 1+\theta_n \tag{6}$$

where $|\theta_n| \leq \gamma_n$ with γ_n defined by

$$\gamma_n := \frac{nu}{1-nu}\,. \tag{7}$$

This notation allows us to count and bound rounding errors; θ_j means that j rounding errors have occurred. Typically θ_j does not refer to a *specific* set of j rounding errors; one θ_3 is not the same as another θ_3, typically, and this must be kept in mind. However, once one gets used to it, it's surprisingly simple and informative.

Theorem 3.1. *For $0 \leq s \leq 1$ and with real Taylor coefficients p_j and q_j, Algorithm 1 produces the exact sum for slightly different Taylor coefficients, namely $\mathrm{fl}(p_j) = p_j(1+\theta_{3m-j+n+4})$ and $\mathrm{fl}(q_j) = q_j(1+\theta_{3n-j+m+4})$. This implies that this algorithm applied to a blend produces the exact value of a blend with coefficients differing at most by a factor $1+\gamma_{\max(3m+n,3n+m)+4}$ from their inputs.*

This is a *componentwise relative* backward error result, and in particular, zero coefficients are not disturbed. As such, it is quite a strong statement; *cf.* the bounds (some normwise) in [14].

Proof. Take the loops in turn. In the interest of readability, we will start to execute the loops "by hand" in order to see the patterns. First, $a_0 \leftarrow 1$ has no rounding error. Next,

$$a_1 \leftarrow \frac{n+k}{k} \cdot \sigma \cdot a_0 \tag{8}$$

gets *three* rounding errors (not four, because addition of integers small enough to be represented as "flints" or "floating-point integers" incurs no rounding error): one for the division and two for the

[3]Maple is compliant with this radix-independent standard, which is described for instance at https://en.wikipedia.org/wiki/IEEE_854-1987.

Algorithm 1 Half Sum, without derivatives

```
1:  procedure HALFSUM(σ, m, n, w)          ▷ m,n nonnegative integers, w Array(0,m)
2:      a_0 ← 1                            ▷ Accumulator loop
3:      for k from 1 to m do               ▷ n gets used in the sum
4:          a_k ← (n + k) · σ · a_{k−1}/k  ▷ 0 ≤ σ ≤ 1
5:      end for
6:      s_0 ← a_0                          ▷ s_k are partial sums of nonnegative numbers
7:      for k from 1 to m do
8:          s_k ← s_{k−1} + a_k
9:      end for
10:     U ← 0
11:     for j from m by −1 to 0 do         ▷ Coefficients w_j used here
12:         U ← s_{m−j} · w_j + σ · U
13:     end for
14:     C ← 1
15:     for j from 1 to n + 1 do           ▷ The complementary factor is (1 − σ)^{n+1}
16:         C ← (1 − σ) · C
17:     end for
18:     S ← C · U
19:     return S                           ▷ The half sum is S
20: end procedure
```

multiplications. Thus $\mathrm{fl}(a_1) = a_1(1+\delta_{1,1})(1+\delta_{1,2})(1+\delta_{1,3})$ where each $|\delta| < u$, the unit roundoff. By induction we find

$$\begin{aligned}\mathrm{fl}(a_j) &= a_j \prod_{k=1}^{j}(1+\delta_{k,1})(1+\delta_{k,2})(1+\delta_{k,3}) \\ &= a_j\left(1+\theta_{3j}\right)\ . \end{aligned} \tag{9}$$

It will be important in what follows that all a_j and their floating-point computed values are nonnegative.

The second loop, which accumulates the partial sums $s_j = \sum_{k=0}^{j} a_j$, sees the following rounding errors. As before $s_0 = a_0$ incurs no rounding error. The next term

$$\mathrm{fl}(s_1) = (s_0 + a_1(1+\theta_3))(1+\delta_1) \tag{10}$$

begins to accumulate error.

$$\mathrm{fl}(s_2) = (s_0 + a_1(1+\theta_3))(1+\delta_1)(1+\delta_2) + a_2(1+\theta_6)(1+\delta_2)\,. \tag{11}$$

Teasing out the pattern, we find

$$\mathrm{fl}(s_k) = a_0(1+\theta_k) + \sum_{j=1}^{k} a_j(1+\theta_{k+2j+1})\,, \tag{12}$$

which is simple to establish by induction on k. We now use the nonnegativity of the terms. We have

$$\mathrm{fl}(s_k) - s_k = a_0\theta_k + \sum_{j=1}^{k} a_j\theta_{k+2j+1} \tag{13}$$

and taking absolute values and using the triangle inequality we have

$$|\mathrm{fl}(s_k) - s_k| \le s_k \gamma_{3k+1} \tag{14}$$

or what is equivalent, $\mathrm{fl}(s_k) = s_k(1 + \theta_{3k+1})$.

The third loop is the one in which we shall reflect these rounding errors back into the input coefficients w_j. This happens in two ways. First, when any given w_j for $j < m$ first enters the sum, it is multiplied by s_{m-j} and thus picks up a multiplier $(1 + \delta)$ with $|\delta| \le \gamma_{3(m-j)+1}$. It also picks one up from the addition. Each iteration thereafter (there are j of them) adds two more rounding error multipliers, from the addition and the multiplication by sigma. The first w_j to enter in the sum is w_m and is special, because it is multiplied by s_0 which has no rounding error. In the m iterations which follow, it picks up m powers of $(1 + \delta_1)(1 + \delta_2)$ which is equivalent to having started with $w_m(1 + \theta_{2m+1})$. The next term, w_{m-1}, picks up a $(1 + \theta_4)$ from the s_1, a $(1 + \delta)$ from the addition, and $m - 1$ powers of $(1 + \delta_1)(1 + \delta_2)$ thereafter, which is equivalent to having started with $w_{m-1}(1 + \theta_{4+2(m-1)+1}) = w_{m-1}(1 + \theta_{2m+3})$. The next term, w_{m-2}, picks up $(1 + \theta_7)$ from s_2, a $(1 + \delta)$ from the addition, and a factor $(1 + \theta_{2(m-2)})$ from the $m - 2$ remaining iterations. This is equivalent to starting with $w_{m-2}(1 + \theta_{2m+4})$. The pattern is now clear and can be established by induction: all the rounding errors are equivalent to changing w_{m-j} to $w_{m-j}(1 + \theta_{2m+j+2})$. The largest change occurs for w_0 and is θ_{3m+2}.

Finally, we have to multiply by the complementary factor $(1 - \sigma)^{n+1}$. As done in the algorithm, this makes a further $n + 2$ rounding errors, each of which can be put into the w_j by linearity. If this loop were instead executed by binary powering, which it could be, then only $O(\lg n)$ rounding errors would be committed. I leave this refinement to the next upgrade.

Since the complete blend executes this sum again with m and n interchanged and with $\sigma = s$ interchanged with $\sigma = 1 - s$, we see that the backward error is bounded by $\gamma_{\max(3m+n,3n+m)+4}$.

∎

Theorem 3.2. *For complex Taylor coefficients, the componentwise relative backward error in a blend computed by Algorithm 1 is bounded by* $\gamma_{21\sqrt{2}\max(m,n)+14\sqrt{2}}$.

Proof. This is a simple overestimate, based on an overestimate of the worst possible complex floating-point error bound, which is that for division when the division is implemented in a way that avoids overflow. The error in division of complex numbers x and y is bounded by [11, p. 73]

$$\mathrm{fl}\left(\frac{x}{y}\right) = \left(\frac{x}{y}\right)(1 + \delta) \tag{15}$$

where $|\delta| \le \sqrt{2}\gamma_7$. This in turn is smaller than $\gamma_{7\sqrt{2}}$ and this is thus worse than that for any operation. This gives the bound of the corollary. ∎

Given an actual computation, one can use a variation of the Oettli–Prager theorem [5, Ch. 6] to compute the actual backward error. I did this for a blend with $m = 19$ and $n = 32$ with Taylor coefficients randomly chosen from a uniform distribution on $[-1, 1]$, when evaluated at 2023 equally-spaced points on $0 \le s \le 1$. The predicted bound is drawn with a red line in Figure 2, and the computed backward errors (which, naturally enough, are different for every different value of s) are shown as black points. We see that the overestimate of the theorem is not that much of an overestimate. For the bound, one imagines rounding errors as always adding up; but one can instead estimate, and a random walk with n steps can be expected to get a distance $O(\sqrt{n})$ away. If we replace the $3m + n$ by its square root, and likewise $m + 3n$, we sometimes get a fairer estimate of what the actual error is. This is drawn with a blue dashed line in Figure 2.

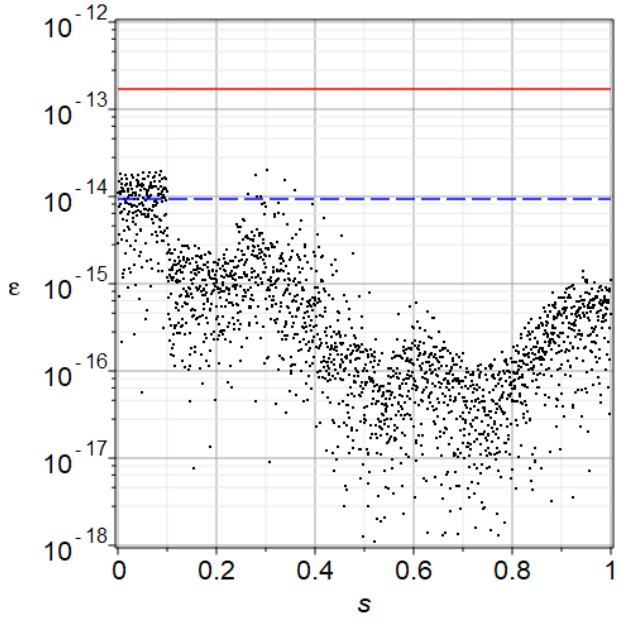


Fig. 2. The actual backward error (black points) versus the theoretical bound (red line) from Theorem 3.1. The blue dashed line is an estimate that arises by replacing $\max(3m+n, m+3n)$ in the bound by its square root, which models rounding errors as being like random numbers.

4 The Lebesgue Function

Let us begin with the basic observation that the Lebesgue function for the blend $H_{m,n}(s)$ is, on $0 \le s \le 1$, simply

$$L_{m,n}(s) = \sum_{j=0}^{m} \left[\sum_{k=0}^{m-j} \binom{n+k}{k} s^{k+j} (1-s)^{n+1} \right] + \sum_{j=0}^{n} \left[\sum_{k=0}^{n-j} \binom{m+k}{k} s^{m+1} (1-s)^{k+j} \right] . \tag{16}$$

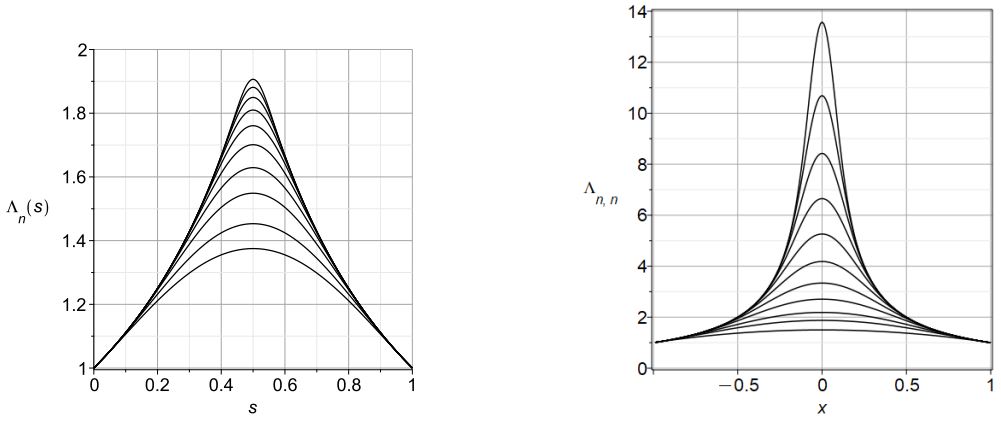
This is just equation (1) with all $p_j = 1$ and all $q_j = (-1)^j$. This follows from the nonnegativity of each of the basis elements on the interval $0 \le s \le 1$.

A similar statement holds for the Bernstein basis $B_k^n(s) = \binom{n}{k} s^k (1-s)^{n-k}$ for $0 \le k \le n$, where the Lebesgue constant is simply 1 (which is optimal). This is because the Bernstein basis is a partition of unity and each basis element is nonnegative on $0 \le s \le 1$. So the two-point Hermite interpolant basis behaviour is not unique, or even the best.

Still, let us demonstrate. In [6] one finds a graph of the first few balanced Lebesgue functions. Figure 3a shows another version, with grades being Fibonacci numbers. The maximum occurs in the center, at $s = 1/2$, and the graph is unimodal. The value of $L_{m,m}(s)$ at $s = 1/2$ is, by computing the first few and then consulting the Online Encyclopedia of Sequences at https://oeis.org/A000346,

$$\Lambda_m\left(\frac{1}{2}\right) = 2 - \frac{1}{2^{2m}} \binom{2m+1}{m+1} . \tag{17}$$

Maple's asymptotics show that this is $2(1 - 1/\sqrt{\pi m}) + O(1/m)$.



(a) On the interval $[0, 1]$

(b) On the interval $[-1, 1]$

Fig. 3. Balanced blend Lebesgue functions $\Lambda_n(s) = L(n, n, s)$ for $n = 2, 3, 5, 8, 13, 21, 34, 55, 89, 144$. On the left, for the interval $[0, 1]$, the approach to 2 in the center as $n \to \infty$ is quite slow, being $2 - 2/\sqrt{\pi n} + O(1/n)$. On the right, for the interval $[-1, 1]$, the growth in the middle seems unbounded, although not very rapid. In fact it grows like $2\sqrt{n/\pi}$.

When one translates to $[-1, 1]$ and investigates unit data there, one finds something different. A few Lebesgue functions for interpolation on $[-1, 1]$ are plotted in Figure 3b. The growth is like $2\sqrt{n/\pi}$, which needs some explanation.

In one sense, nothing has been proved so far in this exposition. In another, everything has. I have used Maple to compute $L_{m,m}(1/2)$ for m up to 1000, which last took about thirty seconds to compute. This verified the above statement for all practical degrees. Blends are useful for high degree, but even so there is a limit.

Nonetheless we should like an actual proof for all m. However, it isn't very simple even to show that the derivative of $L_{m,m}(s)$ is zero at only one place on $0 \le s \le 1$ (which must be $s = 1/2$ by symmetry). A priori, there might be local maxima, or pairs of global maxima. This needs to be ruled out.

The limiting curve, which is $1/(1 - s)$ on $0 \le s \le 1/2$ and $1/s$ on $1/2 \le s \le 1$, has the value 2 when $s = 1/2$, but proving that this is the limiting curve also isn't just a calculus exercise. The blend only just barely converges at $s = 1/2$ (witness the very slow $O(1/\sqrt{m})$ behaviour) because of course this join of two analytic curves is not analytic there. But the Taylor series at each end *do* converge, and this might help.

An alternative would be to try for a combinatorial proof that

$$L_{m,m}\left(\frac{1}{2}\right) = \sum_{j=0}^{m} \sum_{k=0}^{m-j} \binom{m+k}{k} \left(\frac{1}{2}\right)^{m+k+j} = 2 - \frac{1}{2^{2m}} \binom{2m+1}{m+1}. \tag{18}$$

This, together with a proof of unimodality, would do. However, Maple is unable to evaluate this sum (and Maple's abilities are better than mine, with combinatorial sums).

Perhaps unsurprisingly, the answer is already in the best textbook[4] on combinatorics, namely [9]. One can give a nice short proof by using a remarkable sum that can be found on p. 163 in [9], in

[4]Fight me.

their equation 5.20:

$$\sum_{j=0}^{m} \binom{m+k}{k} 2^{-k} = 2^m \,. \tag{19}$$

We now replace the sum to $m-j$ with the sum to m instead, and we get the desired bound.

$$\begin{aligned} L_{m,m}\left(\frac{1}{2}\right) &= \sum_{j=0}^{m} \sum_{k=0}^{m-j} \binom{m+k}{k} \left(\frac{1}{2}\right)^{m+k+j} \\ &\le \frac{1}{2^m} \sum_{j=0}^{m} \frac{1}{2^j} \sum_{k=0}^{m} \binom{m+k}{k} \left(\frac{1}{2}\right)^{k} \\ &\le \frac{1}{2^m} \sum_{j=0}^{m} \frac{1}{2^j} 2^m \\ &\le \sum_{j=0}^{m} \frac{1}{2^j} \\ &\le 2 - 2^{-m} \\ &< 2 \,. \end{aligned} \tag{20}$$

This establishes that $L_{m,m}(1/2) < 2$.

Now all we need to do to complete the proof is to show the "unsurprising" and "obvious"[5] result that the maximum of $L_{m,m}(s)$ must occur in the middle.

There is the fact that $L_{m,m}(s) = L_{m,m}(1-s)$, which means that it is *even* as a function of δ if $s = 1/2 + \delta/2$. This means that the degree is not the same as the grade: $m+m+1$ is the grade, but the degree can be at most $2m$. In other words, the leading coefficient must cancel.

The easiest way to see this last fact, that the leading coefficient must cancel, is to write

$$L_{m,m}(s) = \sum_{j=0}^{m} \sum_{k=0}^{m-j} \binom{m+k}{k} \left(s^{m+1}(1-s)^{k+j} + s^{k+j}(1-s)^{m+1} \right) \tag{21}$$

and to rewrite in the variable $s = 1/2 + \delta/2$:

$$\begin{aligned} s^{m+1}(1-s)^{k+j} + s^{k+j}(1-s)^{m+1} &= \left(\frac{1}{2}+\frac{\delta}{2}\right)^{m+1} \left(\frac{1}{2}-\frac{\delta}{2}\right)^{k+j} + \left(\frac{1}{2}+\frac{\delta}{2}\right)^{k+j} \left(\frac{1}{2}-\frac{\delta}{2}\right)^{m+1} \\ &= \left(\frac{1}{4}-\frac{\delta^2}{4}\right)^{k+j} \left(\left(\frac{1}{2}+\frac{\delta}{2}\right)^{m+1-k-j} + \left(\frac{1}{2}-\frac{\delta}{2}\right)^{m+1-k-j} \right) \\ &= 2^{-(m+1+k+j)} \left(1-\delta^2\right)^{k+j} \left((1+\delta)^{m+1-k-j} + (1-\delta)^{m+1-k-j} \right) \end{aligned} \tag{22}$$

and all the odd-degree terms in δ cancel.

In frustration at not being able to prove such a simple thing, let us try another tack, and try to prove that $L_{m,m}(s) > L_{m-1,m-1}(s)$ on $0 < s < 1$, for all $m \ge 1$. This would mean that each balanced Lebesgue function graph must lie above the previous one, which would explain what we see in Figure 3a. When we compute the first few differences $\Delta_m = L_{m,m}(s) - L_{m-1,m-1}(s)$ we find data that supports the following conjecture:

$$\Delta_m = \frac{1}{m+1} \binom{2m}{m} s^m (1-s)^m \,. \tag{23}$$

[5] One of my old analysis professors used to say that "If something is *obvious*, then you ought to be able to prove it."

The constant in front is a Catalan number, which I identified by using the Online Encyclopedia of Integer Sequences at https://oeis.org/A000108. Proving this surprising fact would be interesting all by itself, and certainly would prove that the Lebesgue function is increasing. Well, we can do this.

THEOREM 4.1. *Equation (23) is true.*

PROOF. The first $m-1$ Taylor coefficients of $L_{m,m}(s)$ and $L_{m-1,m-1}(s)$ at $s=0$ and at $s=1$ are identical, and so the corresponding Taylor coefficients of Δ_m are zero. The degree of Δ_m is at most $2m$, because the degree of $L_{m,m}$ is at most $2m$. Thus the factor $s^m(1-s)^m$ is correct. The value of Δ_m at $s=1/2$, or the value of the mth derivative at $s=0$ or at $s=1$ will then determine the constant in front. If we can show that this is the mth Catalan number, we are done.

We can do this by partial fractions, as follows. First note that $\Delta_m = C_m s^m + O(s^{m+1})$ so identifying the mth Taylor coefficient at $s=0$ suffices[6]. We have

$$\begin{aligned}\frac{\Delta_m^{(m)}(0)}{m!} &= \frac{L_{m,m}^{(m)}(0)}{m!} - \frac{L_{m-1,m-1}^{(m)}(0)}{m!} \\ &= 1 - \frac{L_{m-1,m-1}^{(m)}(0)}{m!}\,,\end{aligned} \tag{24}$$

where we have used the known Taylor coefficient, 1, for $L_{m,m}$ at zero. Next we must differentiate $L_{m-1,m-1}(s)$. Rather than differentiate each term in the formula, we can instead use partial fractions and a contour integral. We write the partial fraction expansion[7] of

$$\begin{aligned}\frac{1}{s^{m+1}(1-s)^m} &= \frac{1}{s^{m+1}}\sum_{k=0}^{m}\binom{m-1+k}{k}s^k + \frac{1}{(1-s)^m}\sum_{k=0}^{m-1}\binom{m+k}{k}(1-s)^k \\ &= \sum_{k=0}^{m}\binom{m-1+k}{k}\cdot\left(\frac{1}{s^{m+1-k}}\right) + \sum_{k=0}^{m-1}\binom{m+k}{k}\left(\frac{1}{(1-s)^{m-k}}\right) \\ &= \sum_{k=0}^{m}\binom{m-1+k}{k}\cdot\left(\frac{1}{s^{m+1-k}}\right) + \sum_{k=0}^{m-1}\binom{m+k}{k}\left(\frac{(-1)^{m-k}}{(s-1)^{m-k}}\right)\,.\end{aligned} \tag{25}$$

Now multiply by any polynomial $p(s)$ of grade $2m-1$ (note that $L_{m-1,m-1}(s)$ is of this grade, and actually of degree $2m-2$) and integrate counterclockwise around a contour that contains $s=0$ and $s=1$, and divide by $2\pi i$. By direct computation we get

$$\frac{1}{2\pi i}\oint \frac{p(s)}{s^{m+1}(1-s)^m}\,ds = 0 \tag{26}$$

because the degree of the denominator is two more than the grade of the numerator. Separating out and using the Cauchy Integral Formula, namely that for a function $f(z)$ analytic inside a contour C containing $z=a$,

$$\frac{f^{(j)}(a)}{j!} = \frac{1}{2\pi i}\oint_C \frac{f(\zeta)}{(\zeta-a)^{j+1}}\,d\zeta, \tag{27}$$

we get

$$\sum_{k=0}^{m}\binom{m-1+k}{k}\cdot\frac{p^{(m-k)}(0)}{(m-k)!} + \sum_{k=0}^{m-1}(-1)^{m-k}\binom{m+k}{k}\cdot\frac{p^{(m-1-k)}(1)}{(m-1-k)!} = 0\,. \tag{28}$$

[6] At this point, the C_m in this formula is just a constant; we don't yet know for sure if it's a Catalan number. But let's use C_m to denote it.

[7] Partial fractions just like these were used in the derivation of the Hermite interpolation formula itself. This means that I was able to write these down very nearly immediately, but the appearance of these binomial coefficients might take an unprepared reader by surprise. They are worth checking, so please feel free!

We take $p(s) = L_{m-1,m-1}(s)$, put in $L^{(m-k)}_{m-1,m-1}(0)/(m-k)! = 1$ for $1 \le k \le m$ and $L^{(m-1-k)}_{m-1,m-1}(1)/(m-1-k)! = (-1)^{m-1-k}$ for $0 \le k \le m-1$ and solve for the mth derivative ($k = 0$ in the first term):

$$\frac{L^{(m)}_{m-1,m-1}(0)}{m!} = -\sum_{k=1}^{m}\binom{m-1+k}{k} + \sum_{k=0}^{m-1}\binom{m+k}{k}. \tag{29}$$

Maple can evaluate these sums in closed form[8], and at last we get

$$\begin{aligned}\frac{\Delta^{(m)}_m(0)}{m!} &= 1 + \sum_{k=1}^{m}\binom{m-1+k}{k} - \sum_{k=0}^{m-1}\binom{m+k}{k} \\ &= 2\binom{2m-1}{m} - \binom{2m}{m+1}\end{aligned} \tag{30}$$

which can be simplified by hand to the Catalan number $C_m = \binom{2m}{m}/(m+1)$. ♮

There is a lovely corollary: Because we know the differences between successive Lebesgue functions, they can be expressed as a single sum.

Corollary 4.2.

$$L_{m,m}(s) = \sum_{k=0}^{m}\frac{1}{k+1}\binom{2k}{k}s^k(1-s)^k\,. \tag{31}$$

Proof.

$$\begin{aligned}L_{m,m}(s) &= L_{0,0}(s) + \sum_{k=1}^{m}\left(L_{m,m}(s) - L_{m-1,m-1}(s)\right) \\ &= 1 + \sum_{k=1}^{m}\Delta_m(s) \\ &= 1 + \sum_{k=1}^{m}\frac{1}{k+1}\binom{2k}{k}s^k(1-s)^k \\ &= \sum_{k=0}^{m}\frac{1}{k+1}\binom{2k}{k}s^k(1-s)^k\,.\end{aligned} \tag{32}$$

♮

In this form, it is obvious[9] that the maximum occurs at $s = 1/2$ (which we have been trying to prove forever, it seems) and moreover we have an explicit sum for it, that Maple can evaluate in closed form:

$$\Lambda_m = \sum_{k=0}^{m}\frac{1}{(k+1)2^{2k}}\binom{2k}{k} = 2 - \frac{1}{2^{2m+1}}\binom{2m+2}{m+1}. \tag{33}$$

At this point one's brain goes "Generating function for the Catalan numbers!" and indeed we will see this explicitly, later. After a little algebra, we can see that this is the same as equation (17).

Corollary 4.3. *Define $F(s) = 1/(1-s)$ for $0 \le s \le 1/2$ and $F(s) = 1/s$ for $1/2 \le s \le 1$. Then*

$$\lim_{m\to\infty} L_{m,m}(s) = F(s) \tag{34}$$

uniformly on $0 \le s \le 1$ and monotonically on $0 < s < 1$.

[8] These ones are easy. I can do them, too. They are direct consequences of equation 5.9 on page 159 of [9] in their section on "Basic Identities".

[9] This time, really obvious: $s = 1/2$ is the point of maximum value of every term.

Proof. Because $\Delta_m(s) > 0$ on $0 < s < 1$ the monotonicity is clear. For uniform convergence we have $0 < F(s) - L_{m,m}(s) \le F(1/2) - \Lambda_m$ for all $0 \le s \le 1$. As previously noted, this goes to zero like $2/\sqrt{\pi m}$. ♮

This "proof" is missing one particular, namely that $L_{m,m}(s) < F(s)$. But we may remedy this as follows. The ordinary generating function for the Catalan numbers [9, p. 358] may be written as

$$\frac{2}{1+\sqrt{1-4x}} = \sum_{n\ge 0} C_n x^n \,. \tag{35}$$

If we put $x = s(1-s)$ and simplify the left hand side in Maple we get

$$\frac{2}{1+\mathrm{csgn}(2s-1)\,(2s-1)} \tag{36}$$

where the `csgn(z)` function is 1 if $\Re(z) > 0$; also if $\Re(z) = 0$ but $\Im(z) > 0$, and -1 otherwise if $z \ne 0$, and is undefined at $z = 0$. Putting in 1 for the csgn we get $1/s$, while if we put in -1 we get $1/(1-s)$. The monotone convergence of the generating function if $|x| < 1/4$ is clear by the ratio test; and we have shown directly that it converges if $s = 1/2$ and hence $x = 1/4$.

4.1 Nearly balanced blends: $n = m+1$ or $n = m-1$

Experimentation in Maple shows that

$$L_{m,m+1}(s) - L_{m-1,m}(s) = L_{m+1,m}(s) - L_{m,m-1}(s) = C_{m+1}s^{m+1}(1-s)^{m+1} \tag{37}$$

where again C_n is the nth Catalan number. We therefore conclude that *nearly* balanced blends are just as good. The explicit expression that we get on summing these differences is not the same as before, though, because

$$L_{1,0}(s) = L_{0,1}(s) = \frac{5}{4} - \left(s - \frac{1}{2}\right)^2 \,. \tag{38}$$

Therefore,

$$L_{m,m+1}(s) = L_{m+1,m}(s) = L_{0,1}(s) + \sum_{k=1}^{m} C_{k+1}s^{k+1}(1-s)^{k+1} \,. \tag{39}$$

Nonetheless the maximum is still in the center, and by inspection the value is the same: 2, being approached as $5/4 + 3/4 - 2\binom{2m+4}{m+2}4^{-m-2}$.

This is because the grade of $L_{m,m+1}(s)$ is $2m+2$ which is *even*, which means the degree can be $2m+2$ as well; and thus we get the same approximation properties as the balanced $L_{m+1,m+1}(s)$.

4.2 Translation to $[-1, 1]$

Lebesgue functions are usually standardised to the interval $[-1, 1]$ and to fairly compare to Chebyshev or equispaced interpolational bases, we should do the same. It's easy enough to throw in the appropriate factors of 2, but I found it just as easy to start afresh. In this section, I will write $H_{m,n}^{[-1,1]}(t)$ with $-1 \le t \le 1$ for the two-point Hermite interpolational polynomial matching series at $t = -1$ and at $t = 1$, and $L_{m,n}^{[-1,1]}(t)$ for the Lebesgue function which has series data all coefficients 1 at $t = -1$ and oscillating coefficients $(-1)^j$ at $t = 1$. In this section, I will write $H_{m,n}^{[0,1]}(s)$ and $L_{m,n}^{[0,1]}(s)$ for the blends on $[0, 1]$. This makes the superscript notation for derivatives a bit crowded, but in my opinion readable: $H_{m,n}^{(k)[-1,1]}(t)$ for the kth derivative of the blend with respect to t.

Explicitly, the blend on $[-1,1]$ is

$$H_{m,n}^{[-1,1]}(t) = \sum_{j=0}^{m} 2^j p_j \sum_{k=0}^{m-j} \binom{n+k}{k} \left(\frac{1+t}{2}\right)^{k+j} \left(\frac{1-t}{2}\right)^{n+1} + \sum_{j=0}^{n} (-2)^j q_j \sum_{k=0}^{n-j} \binom{m+k}{k} \left(\frac{1+t}{2}\right)^{m+1} \left(\frac{1-t}{2}\right)^{k+j} \tag{40}$$

and has $H_{m,n}^{(k)[-1,1]}(-1)/k! = p_k$ for $0 \le k \le m$ and $H_{m,n}^{(k)[-1,1]}(1)/k! = q_k$ for $0 \le k \le n$. The Lebesgue function is

$$L_{m,n}^{[-1,1]}(t) = \sum_{j=0}^{m} 2^j \sum_{k=0}^{m-j} \binom{n+k}{k} \left(\frac{1+t}{2}\right)^{k+j} \left(\frac{1-t}{2}\right)^{n+1} + \sum_{j=0}^{n} 2^j \sum_{k=0}^{n-j} \binom{m+k}{k} \left(\frac{1+t}{2}\right)^{m+1} \left(\frac{1-t}{2}\right)^{k+j} \tag{41}$$

which is, analogously to the $[0,1]$ case, the blend with series coefficients 1 at $t=-1$ and series coefficients $(-1)^j$ at $t=1$. The factors of 2^j reflect the impact of the change in width of the interval.

Experimenting, we find that

$$L_{0,0}^{[-1,1]}(t) = 1 \tag{42}$$

and the differences satisfy

$$L_{m,m}^{[-1,1]}(t) - L_{m-1,m-1}^{[-1,1]}(t) = \frac{1}{2^{2m}} \binom{2m}{m} (t+1)^m (1-t)^m \,. \tag{43}$$

This is different: there are no Catalan numbers here. Writing $L_{m,m}^{[-1,1]}(t)$ as a sum of the differences, as before, we find that

$$L_{m,m}^{[-1,1]}(t) = \sum_{j=0}^{m} \frac{1}{2^{2k}} \binom{2k}{k} (t+1)^k (1-t)^k \tag{44}$$

which has its maximum at $t=0$ because each component term has its maximum there. Maple can sum the result:

$$L_{m,m}^{[-1,1]}(0) = \frac{(2m+1)}{2^{2m}} \binom{2m}{m} \tag{45}$$

and the asymptotics of this are, to leading order, $2\sqrt{m/\pi}$. This matches the plots in Figure 3b very well.

All of this is, so far, experimental mathematics. We need to prove that equation (43) is true. Luckily, the same technique works.

Theorem 4.4. *Equation (43) is true.*

Proof. We proceed the same way, and in fact we can re-use the results of the computation of the mth derivative of $L_{m-1,m-1}^{[0,1]}(s)$, if we use the correct powers of two for the Taylor coefficients to compensate for the fact that that computation was on $[0,1]$. We take $p(s) = L_{m-1,m-1}^{[0,1]}(s)$ in equation (28), put in $L_{m-1,m-1}^{[0,1](m-k)}(0)/(m-k)! = 2^{m-k}$ for $1 \le k \le m$ and $L_{m-1,m-1}^{[0,1](m-1-k)}(1)/(m-1-k)! = (-2)^{m-1-k}$ for $0 \le k \le m-1$ and solve for the mth derivative ($k=0$ in the first term):

$$\frac{L_{m-1,m-1}^{[0,1](m)}(0)}{m!} = -\sum_{k=1}^{m} \binom{m-1+k}{k} 2^{-k} + \sum_{k=0}^{m-1} \binom{m+k}{k} 2^{-1-k} \,. \tag{46}$$

That formula used the s-variable on the interval $[0,1]$. To use this in the t-variable on the interval $[-1,1]$, so $L_{m,m}^{[-1,1]}(t) - L_{m-1,m-1}^{[-1,1]}(t) = B_m(t+1)^m(1-t)^m$, we recognize that $L_{m,m}^{[-1,1](m)}(-1)/m! = 1$ while

the mth Taylor coefficient of the right-hand side is $B_m \cdot 2^m$; we then put in the sum from above and get

$$B_m = \frac{1 + \sum_{k=1}^{m} \binom{m-1+k}{k} 2^{-k} - \sum_{k=0}^{m-1} \binom{m+k}{k} 2^{-1-k}}{2^m} . \tag{47}$$

Maple can simplify both those sums[10] directly, and it is with satisfaction that we find that

$$B_m = 2^{-2m} \binom{2m}{m} . \tag{48}$$

This completes the proof. ♮

What remains is to discover the limiting curve. Letting $m \to \infty$, Maple claims that $L_{m,m}(t) \to 1/\sqrt{t^2}$, which is its way of saying $-1/t$ if $t < 0$ and $1/t$ if $t > 0$. By looking at $(1-4x)^{-1/2}$ which is a generating function for the B_m above, and putting in $x = (1+t)(1-t)/4$ and simplifying, we confirm this.

Notice that the Taylor coefficients of $-1/t = 1/(1-(t+1))$ are all 1 at $t = -1$. Similarly, notice that the Taylor coefficients of $1/t = 1/(1+(t-1))$ are $(-1)^j$ at $t = 1$. The limiting curve therefore has not just a derivative singularity in the middle as in the $[0,1]$ case, but something rather worse.

That this result is so different to the $[0,1]$ case deserves a little more discussion. If we put in Taylor coefficients 2^j and $(-2)^j$ into $H_{m,n}(s)$, then we *ought* to get the same growth at $s = 1/2$ as we did at $t = 0$, and we do. Now the Taylor coefficients at the left are for $1/(1-2s)$ which has a singularity at $s = 1/2$, and for $1/(2s-1)$ on the right (which has a singularity at the same place). So we see that translating the *notion* of a Lebesgue function from one interval to the other is possible, but we will not be comparing the same functions, because "unit data" (which means all coefficients with magnitude 1) means that the functions will be different when we change scales. I found this surprising.

If we continue this "compression" process, which is natural in the piecewise polynomial context, say to an interval of width h, we will have functions $1/(1-hs)$ on the left (whose singularity is at $s = 1/h$, a long way away) and $1/(1-h+hs)$ on the right, with singularity at $s = (h-1)/h = 1-1/h$, equivalently far. At $s = 1/2$ the common value is $1/(1-h/2)$ which, for small h, says that the Lebesgue constant will be close to 1. I have never heard such an explanation for the success of piecewise polynomial approximation.

4.3 Unbalanced blends are bad

The above theorem guarantees decent approximant behaviour as $m \to \infty$ for *balanced* blends. When things are *unbalanced*, the Lebesgue function grows rather indecently. As seen in Figure 4 the growth is at least exponential. Evaluating $L_{1,n-1}(-1+2/n)$ we get an expression with $2^{n+2}/(n^2 e)$ as the leading term of its asymptotic behaviour as $n \to \infty$, proving that statement.

5 Integration of a blend

The following construction was given in [6]. Here we will examine some of its numerical properties.

The *definite* integral of a blend over the entire interval will allow us to construct a new blend whose value at any point is the *indefinite* integral of the original blend up to that point, $F(x) = \int_{s=0}^{x} H_{m,n}(s)\, ds$.

[10] The reader may be unused to trusting a computer algebra system to get finite sums reliably. There has been significant research over the past fifty years on this subject, comparable to the work done on integration in finite terms, and the current state of the art of software is remarkably advanced and trustworthy. See for instance [13]. But, if you want to, you can use the remarkable sum from p. 163 of [9] already mentioned, to prove this yourself.

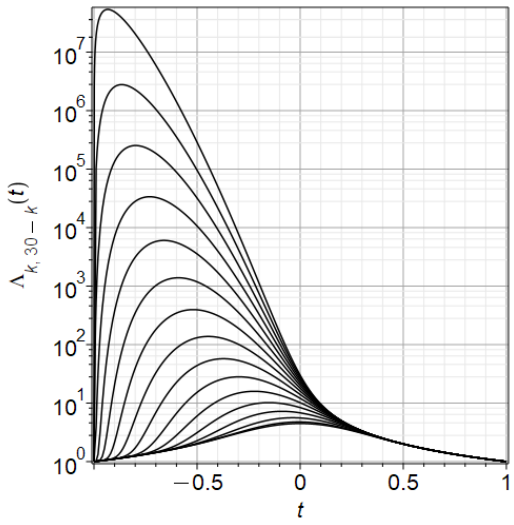


Fig. 4. Lebesgue functions $L_{k,30-k}(t)$ for $k = 0, 1, 2, \ldots, 15$. The most unbalanced, with $k = 0$, has the largest magnitude. We see clearly that this growth is exponential.

Direct integration over the entire interval $0 \le s \le 1$ and use of the formula

$$\int_{s=0}^{1} s^a (1-s)^b \, ds = \frac{a!\; b!}{(a+b+1)\,!}$$

gets us a formula for $F(1)$ involving the symbolic sum

$$\sum_{k=0}^{m-j} \frac{\binom{n+k}{k}\,(j+k)\,!\;(n+1)\,!}{(n+2+j+k)\,!} \tag{49}$$

and a similar one interchanging m and n. Maple can evaluate both those sums:

```
sm := sum( binomial(n+k,k)*(n+1)!*(k+j)!/(j+k+n+2)!, k=0..m-j ):
simplify( sm );
```

yields the right-hand side of the equation below:

$$\sum_{k=0}^{m-j} \frac{\binom{n+k}{k}\,(j+k)\,!\;(n+1)\,!}{(n+2+j+k)\,!} = \frac{(n+m-j+1)\,!\;(1+m)\,!}{(j+1)\,(n+2+m)\,!\;(m-j)\,!}\,. \tag{50}$$

Similarly we find the other sum, and finally we get

$$\begin{aligned}\int_{s=0}^{1} H_{m,n}(s)\,ds = &\frac{(m+1)\,!}{(m+n+2)\,!}\sum_{j=0}^{m}\frac{(n+m-j+1)\,!}{(j+1)\,(m-j)\,!}\,p_j \\ &+\frac{(n+1)\,!}{(m+n+2)\,!}\sum_{j=0}^{n}\frac{(n+m-j+1)\,!}{(j+1)\,(n-j)\,!}\,(-1)^j\,q_j\,.\end{aligned} \tag{51}$$

This is an exact complete integral across the subinterval, if the coefficients are known exactly, but the main use of this routine is when the coefficients are floating-point numbers, in which case this becomes a kind of numerical quadrature. From this formula, one can construct an exact blend for the indefinite integral across the blend because now the Taylor coefficients *for the integral* are known at each end.

5.1 Accuracy and stability

Because the formula has been principally of use for me with floating-point approximations to the Taylor coefficients, I found the following theorem to be useful. Here we use the relation between $H_{m,n}(s)$ and $L_{m,n}(s)$, the Lebesgue function. Specifically, suppose that the Taylor coefficients at $s = 0$ are p_j, $0 \le j \le m$, and the Taylor coefficients at $s = 1$ are q_j, $0 \le j \le n$. Then define the ∞ norm of these vectors as

$$\|\mathbf{p}\|_\infty := \max_{0\le j\le m} |p_j| \tag{52}$$

and similarly for $\mathbf{q}$. We also suppose that each of these vectors may be perturbed to $p_j(1+\delta p_j)$ and $q_j(1+\delta q_j)$ respectively. We may write $\Delta p_j = p_j \delta p_j$ and $\Delta q_j = q_j \delta q_j$ to convert between absolute and relative errors. Then we are interested in bounding the change in the integral that is induced by these changes in the coefficients.

As before the Lebesgue function is the blend that arises when all coefficients of $\mathbf{p}$ are 1 while all coefficients of $\mathbf{q}$ are $(-1)^j$ for $0 \le j \le n$.

By linearity we have

$$\begin{aligned}\int_{s=0}^1 H_{m,n}(s; \mathbf{p}+\Delta\mathbf{p}, \mathbf{q}+\Delta\mathbf{q})\, ds &= \int_{s=0}^1 H_{m,n}(s;\mathbf{p},\mathbf{q})\, ds + \int_{s=0}^1 H_{m,n}(s;\Delta\mathbf{p},\Delta\mathbf{q})\, ds \\ &= I + \Delta I,\end{aligned} \tag{53}$$

say. Then

$$\begin{aligned}|\Delta I| &\le \int_{s=0}^1 |H_{m,n}(s;\Delta\mathbf{p},\Delta\mathbf{q})|\, ds \\ &\le \int_{s=0}^1 L_{m,n}(s)\, ds \max(\|\Delta\mathbf{p}\|_\infty, \|\Delta\mathbf{q}\|_\infty)\,.\end{aligned} \tag{54}$$

We have just proved the following theorem.

Theorem 5.1. *If the coefficients of the blend are in error by at most Δp_j for $0 \le j \le m$ and Δq_j for $0 \le j \le n$, then the error in the integral of the blend is bounded by $\int_0^1 L_{m,n}(s)\, ds \max|\Delta p_j|, |\Delta q_j|$.*

Further, the integral can be explicitly computed: The integrals have all been done, and Maple can simplify the sums if $p_j = 1$ and $q_j = (-1)^j$. We find

$$\int_0^1 L_{m,n}(s)\, ds = 2\Psi(n+m+3) - \Psi(m+3) - \Psi(n+3) + \frac{n+m+4}{(n+2)(m+2)}\,. \tag{55}$$

Here $\Psi(n+1) = -\gamma + \sum_{k=1}^n 1/k$ is the logarithmic derivative of the factorial function.

If either n or m goes to infinity while the other remains fixed, this integral grows like $\ln n$ or $\ln m$. If both $n = m$ go to infinity together the integral is asymptotic to $2\ln 2 - 1/(2m) + O(1/m^2)$. Remember that $\ln 2 = 0.6931\ldots$ is smaller than 1, so this fits with our bound earlier.

This shows that for balanced blends, the computation of the integral by using this formula is numerically stable, and that is certainly what is observed in practice.

5.2 Translation to $[-1, 1]$

If we integrate the Lebesgue function for $[-1, 1]$ instead, we get the following.

$$\int_{t=-1}^1 L_{m,n}(t)\, dt = \frac{m+1}{m+n+2} F\left(\begin{matrix}1,1,-m\\2,-n-m-1\end{matrix}\middle|\, 2\right) + \frac{n+1}{m+n+2} F\left(\begin{matrix}1,1,-n\\2,-n-m-1\end{matrix}\middle|\, 2\right) \tag{56}$$

where F is a hypergeometric function. Since some of the numerator indices are negative, those are just polynomials and this is in effect a restatement of the original sum. When $m = n$, we get an

expression that does not contain hypergeometric functions, namely

$$\int_{t=-1}^{1} L_{m,m}(t)\,dt = \frac{\Psi\left(m+\frac{3}{2}\right)}{2} + \frac{\gamma}{2} + \ln(2)\ . \tag{57}$$

Asking for the asymptotics by **asympt** we get

$$\int_{t=-1}^{1} L_{m,m}(t)\,dt = \frac{1}{2}\ln m + \ln 2 + \frac{\gamma}{2} + O(\frac{1}{m})\,. \tag{58}$$

We therefore see that there is a modest amplification of absolute error if we are using balanced blends integrating across the interval $[-1,1]$ instead of $[0,1]$.

Repeating the computation with $n = m \pm 1$ or even $n = m \pm 2$ gets an expression with the same asymptotics.

If, however, m and n are very different, this amplification factor can get very large indeed, as we saw with the magnitude of $L_{m,n}(t)$ itself. If, say, $m = 5$ and $n = 30$, the factor is more than 2500; whereas if $n = 60$, the factor is $2.75 \cdot 10^{10}$. This looks like exponential growth. The use of this formula for *unbalanced* blends of high degree is therefore not recommended.

6 Build-a-blend: how not to compute derivatives

If you have a Taylor polynomial for a function, say $f(x) \approx f(a)+f'(a)(x-a)+f''(a)(x-a)^2/2$, and you want to compute a Taylor polynomial for $f'(a)$, then you already have one: $f'(x) \approx f'(a)+f''(a)(x-a)$. Of course, it's not as precise an approximation of $f'(x)$ as the original was of $f(x)$, and that might be a problem. With a Taylor polynomial, though, there's not much that can be done.

With a blend, it's somewhat different. There is more information about the function, and for a blend at least that next derivative won't be zero. So we might be tempted to compute $H'''(a)$ from the blend and use it for the 2nd derivative of our approximation to $f'(x)$, adjusting the factorials.

If what you *want* is the derivative of the *blend* and not the derivative of the underlying function, this is actually a technically correct procedure (as it was for the Taylor polynomial, of course). There are situations where this is indeed what is wanted, such as computing the residual of an approximate solution in a differential equation. But even so there are problems, as we will see.

We can use the partial fraction technique that we used to prove Theorem 4.1 for this, as follows.

$$\begin{aligned}\frac{1}{s^{m+2}(1-s)^{n+1}} &= \frac{1}{s^{m+2}}\sum_{k=0}^{m}\binom{n+k}{k}s^k + \frac{1}{(1-s)^{n+1}}\sum_{k=0}^{n}\binom{m+1+k}{k}(1-s)^k \\ &= \sum_{k=0}^{m}\binom{n+k}{k}\cdot\left(\frac{1}{s^{m+2-k}}\right) + \sum_{k=0}^{n}\binom{m+1+k}{k}\left(\frac{1}{(1-s)^{n+1-k}}\right) \\ &= \sum_{k=0}^{m}\binom{n+k}{k}\cdot\left(\frac{1}{s^{m+2-k}}\right) + \sum_{k=0}^{n}\binom{m+1+k}{k}\left(\frac{(-1)^{n+1-k}}{(s-1)^{n+1-k}}\right)\ .\end{aligned} \tag{59}$$

As before, multiply by a polynomial $p(s)$ of grade at most $m+n+1$, so that the denominator degree is at least 2 larger. Then integrate over a contour containing both 0 and 1, divide by $2\pi i$ and use the Cauchy Integral Formula. One can isolate the $(m+1)$st Taylor coefficient of $p(s)$ from the others (which are all the Taylor coefficients from 0 up to m at $s = 0$ and all the Taylor coefficients from 0 up to n at $s = 1$) to get

$$\frac{p^{(m+1)}(0)}{(m+1)!} = -\sum_{k=1}^{m+1}\binom{n+k}{k}\frac{p^{(m+1-k)}(0)}{(m+1-k)!} + \sum_{k=0}^{n}\binom{m+1+k}{k}\frac{(-1)^{n-k}p^{(n-k)}(1)}{(n-k)!}\ . \tag{60}$$

With this new derivative, one throws away $p(0)$ and $p(1)$ and builds a new blend out of $p'(0)$, $p''(0)/1!$ (so we have to multiply the old Taylor coefficient by 2), $p^{(3)}(0)/2!$ (so we have to multiply

by 3), and so on up to and including this new $p^{(m+1)}/(m+1)!$, together with $p'(1)$, $p''(1)$, and so on up to $p^n(1)/n!$. Counting carefully, this gives us a grade $m+n-1+1 = m+n$ polynomial to represent the derivative of the original blend, which was of grade $m+n+1$. This is enough! The blend we get will be the exact polynomial that is the derivative of the original polynomial blend.

An analogous formula holds for the $(n+1)$st derivative at the other end, so we could have instead used that one. Perhaps one should alternate, and use the other end for the second derivative.

But I doubt that this is a good idea for high grade blends. All those binomial coefficients! Unless we can do as we did in the Horner's method algorithm for evaluating the blend, and gradually build them up as we go, there is likely to be quite severe difficulties from adding many large numbers with opposite signs. But perhaps high grade blends can be made to work by something like the Horner technique.

But why make the effort, when the semiautomatic derivative is already available? The formula might have theoretical purposes, of course.

7 Concluding remarks

There is more to say, numerically, about blends. One wants to find roots of blends, and there are several techniques. The method of taking derivatives, which I have called "semi-automatic differentiation" elsewhere, could be compared in more detail to the dubious "build-a-blend" technique of the last section. That last technique has never been implemented, so far as I know, although a version of it has been used to construct so-called "differentiation matrices", mostly for use in an inverse fashion [1]. I suspect that because of the presence of large binomial coefficients, which also give trouble for one of the rootfinding methods, the numerical stability of that approach will be inferior to semiautomatic differentiation of a stable method for evaluation, which is what I did implement. Still, the technique will have theoretical uses, as indeed it did in this present paper for Theorem 4.1.

A full numerical analysis of the semiautomatic technique would also be interesting. Experimentally, it performs well. It would be good, however, to have a theorem guaranteeing that good behaviour.

Finally, the final experiments of section 5 show that using the quadrature formula[11] for very unbalanced blends, that is with one of m or n being much larger than the other, to integrate over the interval $[-1,1]$ is a potentially bad idea numerically. However, the theorems show that for balanced blends they are quite good; only $O(\sqrt{n})$ amplification of error for evaluation, and (even better) only $O(\ln n)$ amplification of error for quadrature. Perhaps "decent" is the right word.

Acknowledgments

This is an outgrowth of work done jointly with Erik Postma, and with Mair Zamir & Chris Brimacombe. Again, I thank John C. Butcher for teaching me the contour integration technique for polynomial interpolation, which proved its utility yet again in this paper. I thank Annie Cuyt for comments on an earlier draft. I also thank Nick Trefethen for encouragement and inspiration. The connection with "smoothstep" and CAGD was pointed out by Mathstodon user @tpfto@mathstodon.xyz.

This work was supported by NSERC under RGPIN-2020-06438 and by the grant PID2020-113192GB-I00 (Mathematical Visualization: Foundations, Algorithms and Applications) from the Spanish MICINN.

[11] I would like to learn if this quadrature formula was known. I suspect that Hermite *must* have known it.

References

[1] Amirhossein Amiraslani, Robert M. Corless, and Madhusoodan Gunasingam. Differentiation matrices for univariate polynomials. *Numerical Algorithms*, 83(1):1–31, 2020.

[2] Chris Brimacombe, Robert M. Corless, and Mair Zamir. Computation and applications of Mathieu functions: A historical perspective. *SIAM Review*, 63(4):653–720, January 2021.

[3] Chris Brimacombe, Robert M. Corless, and Mair Zamir. Elliptic cross sections in blood flow regulation. *in preparation*, 2023.

[4] J.M. Carnicer, Y. Khiar, and J.M. Peña. Optimal stability of the Lagrange formula and conditioning of the Newton formula. *Journal of Approximation Theory*, 238:52–66, February 2019.

[5] Robert M. Corless and Nicolas Fillion. *A graduate introduction to numerical methods*. Springer-Verlag, 2013.

[6] Robert M. Corless and Erik J. Postma. Blends in Maple. In *Communications in Computer and Information Science*, pages 167–184. Springer International Publishing, 2021.

[7] 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. Citeseer, 2004.

[8] Rida Farouki and Tim Goodman. On the optimal stability of the Bernstein basis. *Mathematics of computation*, 65(216):1553–1566, 1996.

[9] Ronald L Graham, Donald E Knuth, and Oren Patashnik. *Concrete mathematics: a foundation for computer science*. Addison–Wesley, 1989.

[10] Charles Hermite. *Cours d'analyse de l'École polytechnique*, volume 25. Gauthier-Villars, 1873.

[11] Nicholas J Higham. *Accuracy and stability of numerical algorithms*. SIAM, 2002.

[12] B.Ali Ibrahimoglu and A. Cuyt. Sharp bounds for Lebesgue constants of barycentric rational interpolation. *Exp. Math.*, 25:347–354, 2016.

[13] Manuel Kauers and Peter Paule. *The Concrete Tetrahedron*. Springer, 2011.

[14] Alicja Smoktunowicz. Backward stability of Clenshaw's algorithm. *BIT Numerical Mathematics*, 42(3):600–610, 2002.

[15] Lloyd N Trefethen. *Approximation Theory and Approximation Practice*. SIAM, 2019.

Received 31 January 2023