# An Hermite–Obreshkov Method for the Mathieu Equation

Robert M. Corless

February 2023

Western University, Canada

ANODE 2023, Celebrating the 90th birthday of John C. Butcher

# Maple Transactions

**Announcing Maple Transactions**

a new open access journal with no page charges

mapletransactions.org

# What this talk is about

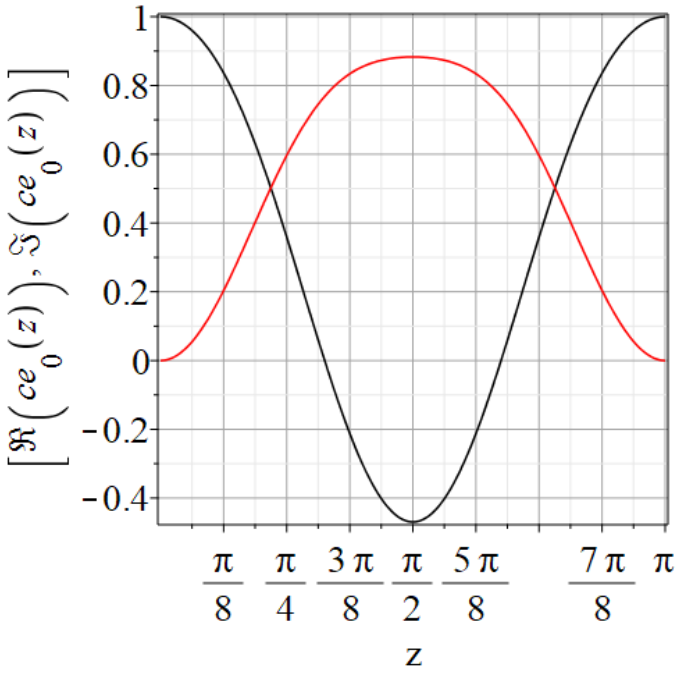



**Figure 1:** A Mathieu function (double eigenvalue) using my code

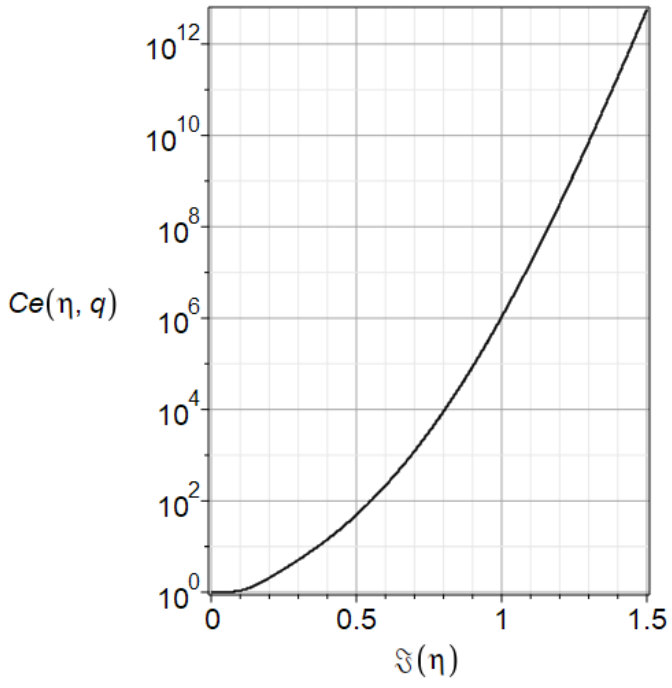
I wrote a collection of routines in Maple for solving the Mathieu differential equation (and other routines for solving the Mathieu eigenvalue problems, based on the method of Gertrude Blanch).

# The Mathieu equation

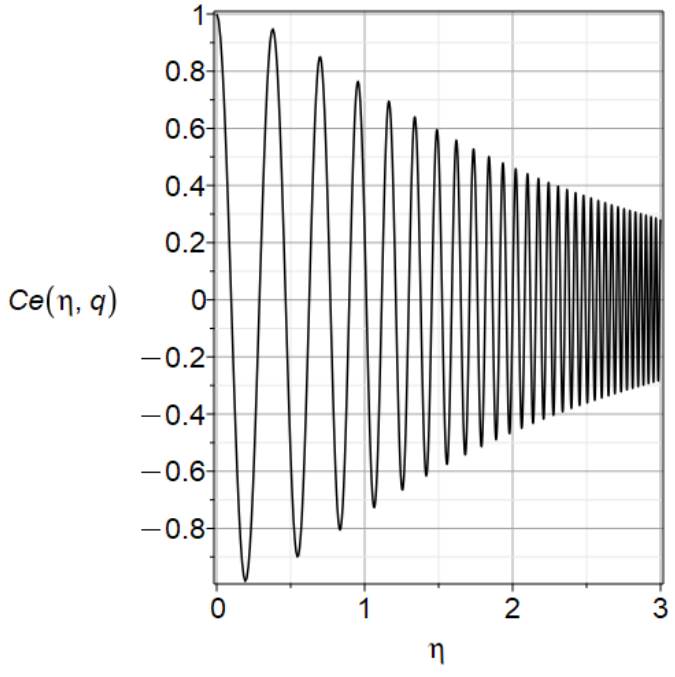
$$\frac{d^2y}{dz^2} + (a - 2q\cos 2z)y = 0 \tag{1}$$

1. $y(0) = 1$ with $y'(0) = 0$ gives $w_I(z)$ A&S, `MathieuC` in Maple
2. $y(0) = 0$ with $y'(0) = 1$, gives $w_{II}(z)$, `MathieuS` in Maple.
3. If $0 \leq z \leq 2\pi$ and $a$ is such that $y(z)$ is periodic then $y(z)$ is a "Mathieu function". In this case $a$ is an eigenvalue for the parameter $q$.
4. Pure imaginary $z$ gives a so-called "modified" Mathieu function.
5. The parameter $q$ depends on the situation being modelled.
6. The desired solution can be highly oscillatory or of *doubly* exponential growth.

# An example




**(a)** Doubly exponential growth




**(b)** Exponentially increasing frequency

**Figure 2:** The solutions of $y'' - (a - 2q\cosh\eta)y = 0$ can grow *doubly* exponentially and oscillate with exponentially increasing frequency; sometimes both

# But why write my own solver?

- All Mathieu codes I knew of could not handle the double eigenvalue problem explicitly.
- I wanted an independent method whose solutions could be verified *a posteriori*
- We had a specific application in mind (blood flow in a vessel with elliptic cross section) where solution for complex $q$ was needed [1]
- The Maple model for evaluation of functions is inefficient for evaluating functions at lots of points at once
- The multiple-precision features of Maple are convenient

# Could there be other uses for this?

- Efficient high-accuracy solution of IVP (or BVP) for "D-finite" or "holonomic" ODE
- Solving delay differential equations (need discontinuity handling, though)
- Ned Nedialkov and John Pryce have already implemented a similar method in a quite general way, for solving DAE. Their code `DAETS` works well. Perhaps some experimental features of this Maple code could influence future development of `DAETS`.
  `http://www.cas.mcmaster.ca/~nedialk/daets/`

# Ancillary Code

To implement this code, I first wrote (with Erik Postma) an **efficient** and **numerically stable** evaluator for what we call "blends" (arbitrary degree two-point Hermite interpolants), together with routines for manipulating them: integration, differentiation, rootfinding, addition, multiplication, etc. [3]

A "string of blends" is a particular kind of piecewise polynomial interpolant, that has some interesting properties, especially in the context of ODE solving.

# Blends

Suppose that we know some Taylor coefficients of a function at two distinct points, say $z = a$ and $z = b$. Then put $z = a + s(b - a)$ and the interval $0 \leq s \leq 1$ determines a line segment in the $z$-plane.

Then (Hermite, Cours d'Analyse 1873)

$$H(s) = \sum_{j=0}^{m} p_j \sum_{k=0}^{m-j} \binom{n+k}{k} s^{k+j} (1-s)^{n+1} + \sum_{j=0}^{n} (-1)^j q_j \sum_{k=0}^{n-j} \binom{m+k}{k} s^{m+1} (1-s)^{k+j} \tag{2}$$

has $H^{(j)}(0)/j! = p_j$ for $0 \leq j \leq m$ and $H^j(1)/j! = q_j$ for $0 \leq j \leq n$. Here differentiation is wrt $s$ so one has to be careful about bookkeeping.

# Numerical stability

1. Blends are *ridiculously* stable numerically. They have been used successfully for degrees up to about 1000.
2. Numerical evaluation of a blend using the double Horner form gives the *exact* value of a blend with Taylor coefficients $p_j(1+\gamma_{O(m)+O(n)})$.
3. The Lebesgue function of balanced blends is bounded by 2 on [0,1] or $2\sqrt{m/\pi}$ on $[-1,1]$, for grade $2m+1$

See RMC "Blends have decent numerical properties" Maple Transactions Vol 3 Issue 1 February 2023,
`https://doi.org/10.5206/mt.v3i1.15890`

# Making it jump

With $p_k = 1,0,0,\ldots$ and $q_k - 1,0,0,\ldots$ and taking $m = 368$ and $n = 631$ (so the grade $m + n + 1 = 1000$):

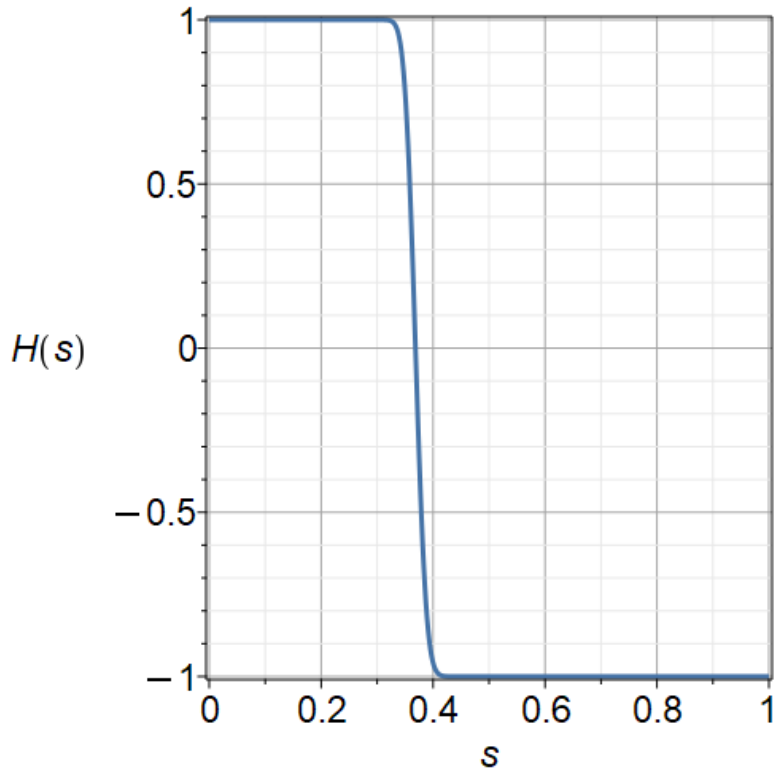



**Figure 3:** Look how smooth the plot is. No "Gibbs phenomenon" at all!

# Blendstrings

A string of blends, or `blendstring`, is a set of the form

$$\mathcal{B} := \{L_k\}_{k=0}^{M} \tag{3}$$

where each $L_k$ is a list of the form

$$L_k := [\alpha_k, C_{k,0}, C_{k,1}, \ldots, C_{k,m_k}] \tag{4}$$

intended to represent the known Taylor coefficients $C_{k,j}$ at the point $z = \alpha_k$.

Two blendstrings are *compatible* if they have the same knots in the same order and with the same degrees $m_k$ at each knot. Then they can be added together, etc.

# A possible blendstring

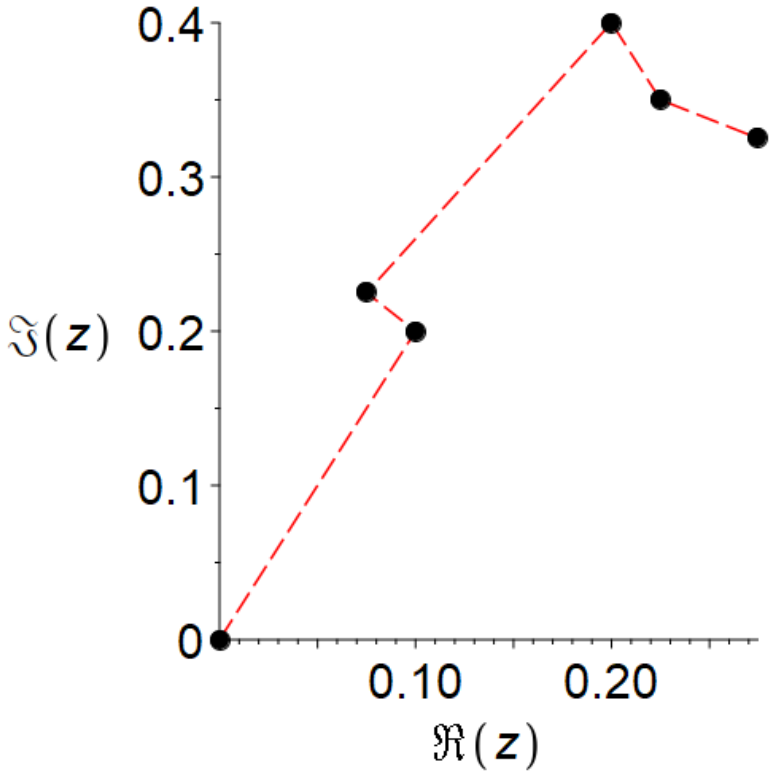



**Figure 4:** The knots $\alpha_k$ are plotted with solid circles. Taylor coefficients are known at those knots. On the segment between any two knots, Hermite's formula gives a "blend" approximating the underlying function. My implementation only allows straight lines of knots so far.

# The solver

Key features (remember it's just for the Mathieu DE):

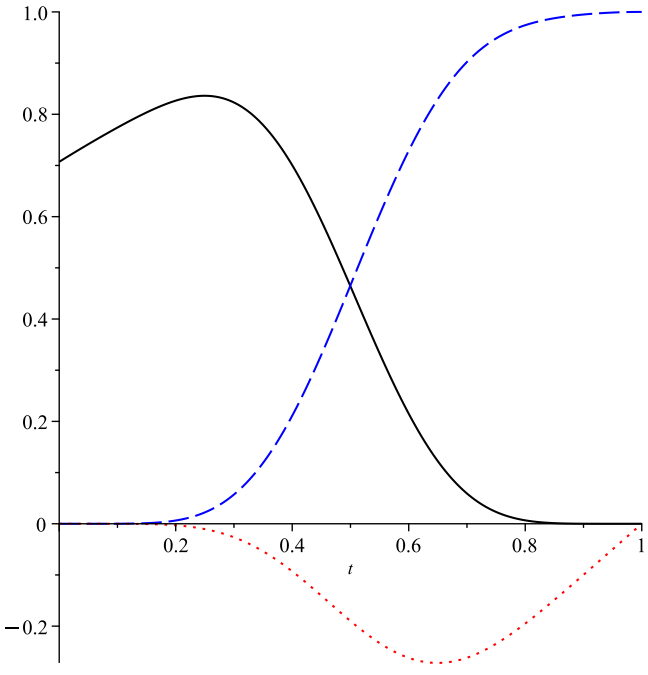
- Fixed order $2m$ that the user can choose by choosing degree $m$ (same at each end); default is 16 by Taylor series of degree 8 at each knot. Hand-coded Taylor series loops. **[Order is not the whole story.]**
- Variable stepsize (uses Gustafsson Lund & Soderlind 1988 PID control)
- Implicit (the Mathieu equation is linear) **Not Stiff; Oscillatory**
- Collocation at $s = 1/4$ and $s = 3/4$ (Mathieu eq is 2nd order)
- Residual measured at $s = 1/2$; "defect control", asymptotically maximum
- The solver returns a blendstring (actually *two*) representing the solution (and the *other* solution, which is sometimes needed and basically free).

# A few details

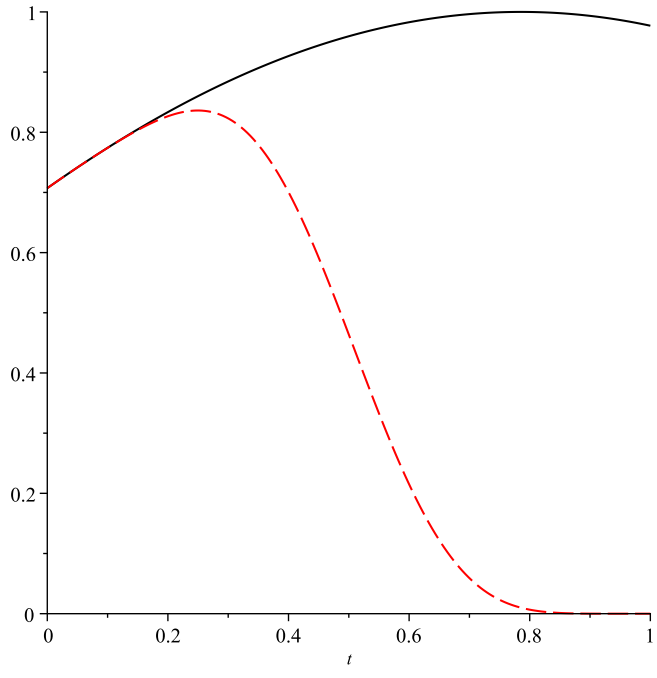
Taking a step from $z_n$ to $z_{n+1} = z_n + \Delta z$ entails:

- Knowing $m+1$ Taylor coefficients at $z = z_n$. Call the solution, blended with 0 at $z = z_{n+1}$, $L(z)$
- Computing $m+1$ Taylor coefficients of $y(z)$ where $y'' + (a - 2q\cos 2z)y = 0$ with $y(z_{n+1}) = 1$ and $y'(z_{n+1}) = 0$. Call the solution, blended with 0 at $z = z_n$, $C(z)$
- Computing $m+1$ coefficients where $y(z_{n+1}) = 0$ and $y'(z_{n+1}) = 1$; call the solution blended with 0 at $z_n$, $S(z)$
- Computing the residuals (defects) of $C$ and $S$ and $L$ at $z = z_n + \Delta z/4$ and at $z = z_n + 3\Delta z/4$ (requires 2nd derivatives of the blends)
- Choosing constants $A$ and $B$ so that the residual of the blend of $AC(z) + BS(z) + L(z)$ is **zero** at both those points
- Testing the residual at $z_n + \Delta z/2$ to see if the step should be accepted; if so, the $m+1$ coeffs of $AC + BS + L$ at $z_{n+1}$ are recorded.

# A picture of the collocation




**(a)** $L$, $C$, and $S$




**(b)** $L$ and $y = A \cdot C + B \cdot S + L$

**Figure 5:** Simple collocation

# Asymptotic Expansion of the Residual

$z = a_n + sh$ so $0 \leq s \leq 1$ on the step $h = a_{n+1} - a_n$

$$\delta(s) = Kh^{2m}s^{m-1}(s - \frac{1}{4})(s - \frac{3}{4})(s-1)^{m-1} + \cdots \tag{5}$$

Max of polynomial occurs at $s = 1/2$ and is $2^{-2m-2}$.

The code applies this error control to both $w_I$ and $w_{II}$ simultaneously. The constant $K$ cannot be zero for both.

# An example

For $q = 1.46876861378514\ldots$ $i$ and $a = 2.0886989027\ldots$ (the Mulholland–Goldstein double eigenvalue) we compute the eigenfunction associated to this double eigenvalue at 15 digit precision, using $m = 10$. Below is the residual using 30 digit precision, verifying *a posteriori*.

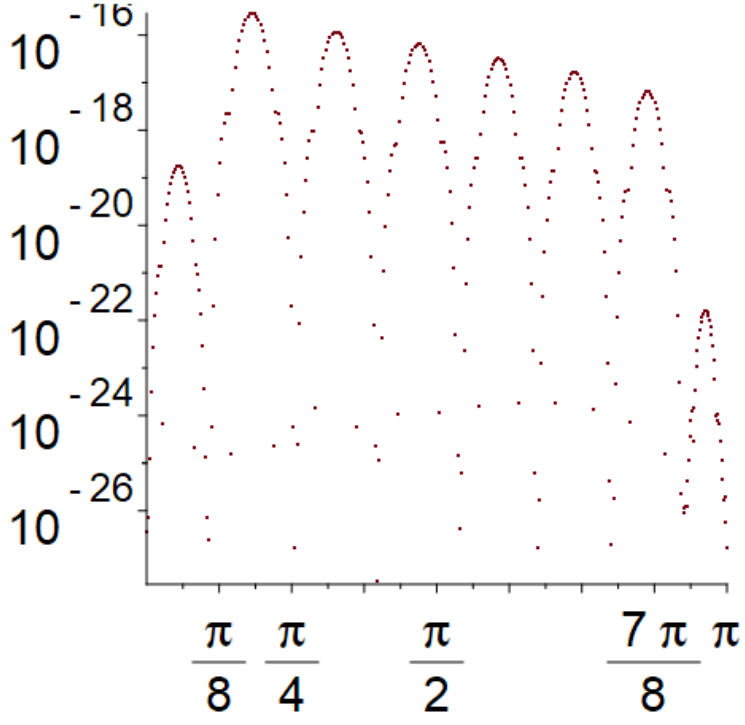



**Figure 6:** Residual computed at preposterously many points

# Quadrature

There is a very interesting (exact!) quadrature formula for a blend, which allows *indefinite* integration of blendstrings. I don't know if this is a reinvention.

$$\int_{s=0}^{1} H(s)\,ds = \frac{(m+1)!}{(m+n+2)!}\sum_{j=0}^{m}\frac{(n+m-j+1)!}{(m-j)!(j+1)}p_j + \frac{(n+1)!}{(m+n+2)!}\sum_{j=0}^{n}\frac{(-1)^j(n+m-j+1)!}{(n-j)!(j+1)}q_j \qquad (6)$$

This is excellent numerically if $n \approx m$. If the blend is very unbalanced, though, the error can be exponentially amplified.

# When $m = n = 2$

When $n = m = 2$ we get the "twice corrected Trapezoidal rule"

$$I(h) = \int_0^h H(s)\,ds = \left(\frac{p_0}{2} + \frac{q_0}{2}\right) h + \left(\frac{p_1}{10} - \frac{q_1}{10}\right) h^2 + \left(\frac{p_2}{60} + \frac{q_2}{60}\right) h^3 \quad (7)$$

This is exact for polynomials of grade 5, so error is $h \cdot O(h^6)$. More precisely,

$$I(h) - \int_0^h f(s)\,ds = -\frac{1}{20}\frac{f^{(7)}(0)}{7!} h^7 + O(h^8)\,. \quad (8)$$

# How John Butcher might derive that formula

Start with the partial fraction expansion

$$\begin{aligned}\frac{1}{s^4(s-h)^4} =& \frac{1}{h^4 s^4} + \frac{4}{h^5 s^3} + \frac{10}{h^6 s^2} + \frac{20}{h^7 s} \\ &+ \frac{1}{h^4\,(s-h)^4} - \frac{4}{h^5\,(s-h)^3} + \frac{10}{h^6\,(s-h)^2} - \frac{20}{h^7\,(s-h)}\,.\end{aligned} \tag{9}$$

Multiply by a polynomial $I(s)$ of grade $6 = 8 - 2$. Then integrate over a contour containing $s = 0$ and $s = h$ (which gives zero because the degree of the denominator is at least greater by 2 than the degree of the numerator) and use the Cauchy Integral formula to obtain the following formula.

# It's so easy

$$I(h) = \left(\frac{D^{(3)}(I)(h)}{120} + \frac{D^{(3)}(I)(0)}{120}\right) h^3 + \left(\frac{D^{(2)}(I)(0)}{10} - \frac{D^{(2)}(I)(h)}{10}\right) h^2 + \left(\frac{D(I)(0)}{2} + \frac{D(I)(h)}{2}\right) h + I(0) \tag{10}$$

With $D(I)(s) = H(s)$ we have the previous formula. The general case can be carried out in exactly the same manner.

(I have not been able to find out if Hermite knew this formula. It certainly seems as if he ought to have. John suggests that Gauss or even Newton might also have known it.)

# Differentiation

The numerical routine Erik Postma and I wrote to evaluate blends also has the ability to evaluate arbitrary derivatives, by what I call "semi-automatic differentiation". Blends are just polynomials, after all. Still, this means that it is easy to differentiate blendstrings, and thus the interpolants from this solver.

# Rootfinding

There is a **companion pencil** for a blend: its eigenvalues are the roots of the polynomial expressed by the blend. Conversion to the monomial basis is avoided. It works well if the degrees are not too large (binomial coefficients *do* bother this method).

Another method is to use Inverse Cubic Iteration [2] (an English version is on the arXiv). This method has convergence order $1 + \sqrt{3} = 2.732$ with the same effort as Newton's method. It was inspired by thinking about low-order blends.

# Example

Suppose $m = 2$ and we know the Taylor coefficients up to the second derivative at $s = 0$ and at $s = 1$. Then the following matrix has $H(s)$ as determinant:

$$\begin{bmatrix} 0 & \frac{p''(1)}{2} & p'(1) & p(1) & \frac{p''(0)}{2} & p'(0) & p(0) \\ -1 & \lambda - 1 & 0 & 0 & 0 & 0 & 0 \\ 3 & -1 & \lambda - 1 & 0 & 0 & 0 & 0 \\ -6 & 0 & -1 & \lambda - 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & \lambda & 0 & 0 \\ 3 & 0 & 0 & 0 & -1 & \lambda & 0 \\ 6 & 0 & 0 & 0 & 0 & -1 & \lambda \end{bmatrix} \tag{11}$$

# Hermite Jump Zeros

If $m = 21$ and $n = 13$, with series 1,0,0,... at the left and $-1, 0, 0, \ldots$ at the right, we get the following zeros using the companion pencil method in double precision. (NB: the very high degree example earlier needs 450 digit multiple precision to find eigenvalues of the companion pencil).

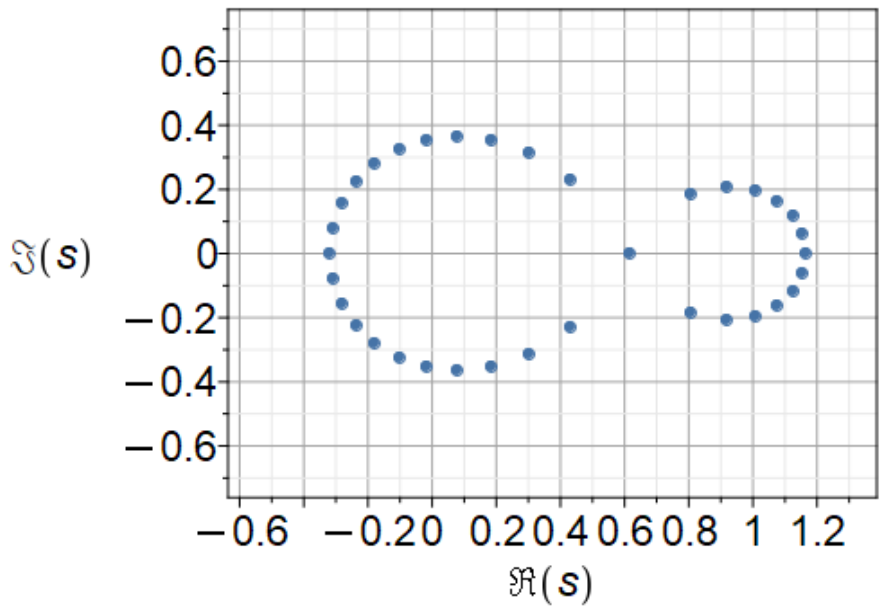



**Figure 7:** Zeros of the grade $21 + 13 + 1 = 55$ polynomial fitting the flat series

# Where next?

The approximation theoretic properties of blends are "understood" in the sense that there are results scattered throughout the literature. By and large they are in many ways inferior even to Chebyshev expansion, much less to the new techniques such as AAA. Nonetheless they seem interesting to me and I shall spend some time writing things down to my satisfaction.

I would also like to explore the singularity detection and location facilities afforded by these interpolants.

Fine-tuning all the safety factors and timeouts in the code should make it more efficient (10% ? more?) (Translating it to a fast language, e.g. Julia, would do a *lot* more)

Extending this to variable order would be fun, but first, generalize it to other IVP.

# Thank you for listening.

This work supported by NSERC, by the Spanish MICINN, and by the Isaac Newton Institute in Cambridge.

I would like to particularly thank Erik Postma and my other co-authors, and especially to thank John C. Butcher for teaching me the contour integral technique for interpolation, by which I (re)derived all these formulae.

**Happy Birthday, John!**

# References

[1] Chris Brimacombe, Robert M. Corless, and Mair Zamir. "Computation and Applications of Mathieu Functions: A Historical Perspective". In: *SIAM Review* 63.4 (Nov. 2021), pp. 653–720. URL: https://doi.org/10.1137/20m135786x.

[2] Robert M Corless. "Sobre la iteración cúbica inversa". In: *Gaceta de la Real Sociedad Matematica Española* 24.1 (2021), pp. 147–159.

[3] Robert M Corless and Erik Postma. "Blends in Maple". In: *arXiv preprint arXiv:2007.05041* (2020).