Analysis versus Algebra in Symbolic Computation

Robert M. Corless
Department of Computer Science
University of Western Ontario
London, CANADA, N5B 4A9
rcorless@uwo.ca

July 2025

The search for formulaic answers via algebra has an ancient history. That such formulæ may have lacunæ where they do not apply was perhaps first noted explicitly by Cauchy in his 1821 *Cours d'Analyse* (English translation presented at the MacTutor site):

> We must even note that they suggest that algebraic formulas have an unlimited generality, whereas in fact the majority of these formulas are valid only under certain conditions and for certain values of the quantities they contain. By determining these conditions and these values, and by fixing precisely the sense of all the notations I use, I make all uncertainty disappear.

But battle was joined anew when digital computers arrived on the scene. The early generations of software performed pretty well all transformations taking an "algebraic" approach and not considering "analytic" issues[1]. This had consequences: the tension between algebra and analysis continues to this day and many current algebra systems will still sometimes give incomplete, misleading, or flatly incorrect answers to various questions.

This paper reviews both progress to date and future prospects in this area. One has to accept that there are unpalatable choices to be made: On the one hand the generation of potentially incorrect results and on the other sometimes prohibitively expensive computations, grossly clumsy presentations of answers or other unfriendly behaviours. Looking at the history and how various systems have responded can help shape an understanding of proper responses to the challenges.

Consider an example from [1] Appendix E, "Symbolic Computation: The Pitfalls." We extract from there the following SymPy (Python 3) code that attempts to compute a conspicuously nonnegative definite integral, namely

$$F = \int_{-1}^{1} \frac{e^{-1/x}}{x^2(1+e^{-1/x})^2}\,dx \ : \tag{1}$$

[1] I'm not being precise, here. Roughly what I intend to convey by "algebra" versus "analysis" is that algebraic models of computation typically have a different notion of continuity than do analytic models of computation, if they consider continuity at all.

```
from sympy import *
x = symbols('x')
import numpy as np
f = exp(-1/x)/(x**2*(1+exp(-1/x))**2)
defint = integrate( f, (x,-1,1))
print('Definite␣integration␣(wrong):␣', simplify(defint))
E = np.exp(1.0)
answe = 1/(1+E) - 1/(1/E+1)
print("That␣numerically␣evaluates␣to␣", answe, "which␣is␣<␣0.")
```

The definite integral answer Python 3.8.16 gives is $1/(1+e) - 1/(1+1/e) = (1-e)/(1+e)$, which is negative, and incorrect.

It is possible that by the time I give my talk, the SymPy definite integral procedure will have been corrected. Things change and progress. Both Maple and Mathematica now get this definite integral correctly (the answer is $2/(1+e)$), but they did not always do so. Of course, the issue is the essential singularity at the origin, which gives rise to a *discontinuous* formula for the antiderivative.

Related but different difficulties occur in solving linear or algebraic equations containing parameters. A comprehensive solution is frequently too lengthy to be useful, but on the other hand a generic answer could miss many cases of interest. See [10], [9], and [2]. Again, the issue is *discontinuity*: many matrix factorings (e.g. Jordan canonical form) and polynomial ideal basis changes (e.g. Gröbner bases) are *discontinuous* at certain sets of parameter values. This is sometimes known as "the specialization problem." Then there is the question of integration of functions containing parameters; to address this, the technique of "Kahanians" is discussed in [6].

The difficulty arises in the symbolic solution of differential equations as well. In the upcoming book [4] we find a discussion of an example from [8, p. 127], namely $\varepsilon y' = x^2(x^2 - y^2)$ subject to $y(-1) = 0$. The solution via Maple 2025 is

```
macro(ep=varepsilon);
de := ep*diff(y(x), x) = x^2*(x^2 - y(x)^2);
sol := dsolve( {de, y(-1)=0}, y(x));
```

$$x\frac{\left(I_{-\frac{5}{8}}\left(\frac{x^4}{4\varepsilon}\right)K_{\frac{5}{8}}\left(\frac{1}{4\varepsilon}\right) - K_{\frac{5}{8}}\left(\frac{x^4}{4\varepsilon}\right)I_{-\frac{5}{8}}\left(\frac{1}{4\varepsilon}\right)\right)}{K_{\frac{3}{8}}\left(\frac{x^4}{4\varepsilon}\right)I_{-\frac{5}{8}}\left(\frac{1}{4\varepsilon}\right) + I_{\frac{3}{8}}\left(\frac{x^4}{4\varepsilon}\right)K_{\frac{5}{8}}\left(\frac{1}{4\varepsilon}\right)} \tag{2}$$

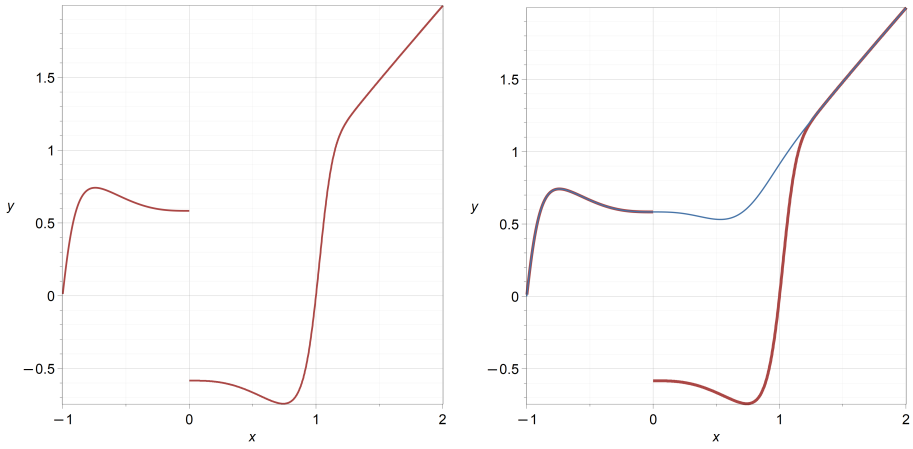
The K and I symbols refer to Bessel functions. O'Malley writes this more neatly, using $\lambda = K(3/8, 4/\varepsilon)/I(-5/8, 4/\varepsilon)$. O'Malley then says, rather cryptically, that this is valid "for a constant λ (which might change at the turning point)." The "turning point" he means is at $x = 0$.

If we plot the solution returned by Maple, equation (2), perhaps by

```
plot( eval(rhs(sol),ep=1/8), x=-1..2 );
```

then the result is seen in figure 1(a) to be *discontinuous* at $x = 0$. Yet the reference analytical solution cannot be discontinuous there (or anywhere). The discontinuity is spurious. If we want a correct formula, it must consist of two parts, one for $x < 0$ and one for $x \geq 0$.

Symbolic computation with discontinuous objects is very difficult. So is numerical computation with discontinuous objects. Incorporating analysis of the discontinuities is the only way to ensure that the answers are correct. Using "provisos" is one lazy way to deal with case splitting [5], and this can be efficient. Knowledge of branch points and branch cuts for elementary and special functions is also necessary [7]. The issue is still not wholly settled.



(a) A discontinuous "solution" (b) Jump corrected

Figure 1: left) A discontinuous putative "solution" computed by Maple that contains discontinuous expressions, namely equation (2) for $\varepsilon = 1/8$. (right) After analyzing the jump at 0, the "constant" of integration can be adjusted to ensure smoothness.

Analogous difficulties with infinite series, especially "divergent" series, also cause difficulty. See [3] for an extended discussion.

In this talk I will describe some of the history of how this battle has unfolded in the symbolic computation community. Some good progress has been made, and some of today's algorithms and implementations are genuinely better than most of those of thirty years ago.

References

[1] Neil J Calkin, Eunice YS Chan, and Robert M Corless. *Computational discovery on Jupyter*. SIAM, 2023.

[2] Changbo Chen and Marc Moreno Maza. Solving parametric polynomial systems by RealComprehensiveTriangularize. In *ICMS 2014: Proceedings 4*, pages 504–511. Springer, 2014.

[3] Robert Corless. Devilish tricks for sequence acceleration. *Maple Transactions*, 3(1), February 2023.

[4] Robert M. Corless and Nicolas Fillion. *Perturbation methods using backward error*. SIAM, Philadelphia, 2025 (anticipated).

[5] Robert M Corless and David J Jeffrey. Well... it isn't quite that simple. *ACM Sigsam Bulletin*, 26(3):2–6, 1992.

[6] Robert M Corless, David J Jeffrey, and David R Stoutemyer. Integrals of functions containing parameters. *The Mathematical Gazette*, 104(561):412–426, 2020.

[7] Robert M Corless, David J Jeffrey, Stephen M Watt, and James H Davenport. "According to Abramowitz and Stegun" or arccoth needn't be uncouth. *ACM SIGSAM Bulletin*, 34(2):58–65, 2000.

[8] Robert E. O'Malley. *Historical Developments in Singular Pertubations*. Springer, 2014.

[9] William Y Sit. An algorithm for solving parametric linear systems. *Journal of Symbolic Computation*, 13(4):353–394, 1992.

[10] Volker Weispfenning. Comprehensive Gröbner bases. *Journal of Symbolic Computation*, 14(1):1–29, 1992.