Symbolic Computation in Nonlinear Dynamics

Robert M. Corless
Department of Applied Mathematics,
The University of Western Ontario,
London, CANADA N6A 5B7
rcorless@uwo.ca

This paper is dedicated to the memory of
M. A. H. Nerenberg (1936–1993).

February 28, 1994

Abstract

This paper gives examples of how computer algebra systems can help us to understand nonlinear dynamical systems and their numerical simulations. We caution against naive use of exact arithmetic, but we give examples where elementary use is helpful. We also look at the use of polynomial computations—such as factoring, computation of discriminants, and Gröbner bases—in bifurcation studies.

1 Introduction

This paper investigates the rôle symbolic computation can play in the understanding of nonlinear dynamical systems. Symbolic computation does not mean 'symbolic dynamics', where the dynamics of the iteration of a map is understood by converting the problem to a shift map on a sequence of 'symbols'. It means, instead, the use of a computer to perform mathematical operations such as exact arithmetic, polynomial manipulations, taking derivatives, and evaluating integrals. We use these facilities here to try to understand certain features of nonlinear systems.

For a representative sample of works using symbolic computation to study nonlinear dynamical systems, see various papers in [2, 14, 18, 21, 23]. This is by no means an exhaustive list.

Overall, symbolic computation can be very useful in symbolic dynamics, but as the first section will show, sometimes it is not helpful at all.

2 Naive Use

The following cautionary tale is included here because I have seen people actually advocate the use of symbolic computation in the following way for nonlinear dynamical systems.

Suppose that we are worried about the effects of floating-point arithmetic on the iteration of a map: say the tent map

$$T(x) = \begin{cases} \mu x & \text{if } x < 1/2 \\ \mu(1-x) & \text{otherwise} \end{cases}$$

or the discrete logistic map

$$L(x) = ax(1-x)\,. \tag{1}$$

This is a reasonable thing to worry about: for certain parameter values, these maps are *chaotic*, which means they are exponentially sensitive to changes in the initial conditions *or to changes in the map*, such as are produced by rounding errors. If you iterate these maps using floating-point arithmetic, in a very short time (say 10 or 15 iterations) the computed orbits will be completely different from the true orbits.

By 'orbits' we mean the set of numbers you get by starting with some given x_0, and producing $x_1 = T(x_0)$, $x_2 = T(x_1)$, $x_3 = T(x_2)$, and so on.

It turns out that for many maps, and in particular for these maps, (for certain parameter values) one can prove that a property called 'shadowing' holds: if your floating-point arithmetic is sufficiently precise, then there is a *true* orbit which is uniformly close to the computed one (in which case we say the true orbit *shadows* the computed one). This may or may not actually help, however. See [11, 15, 22] for a deeper discussion. For now, assume that we want to use *exact* arithmetic to try to avoid these somewhat problematic issues.

Let us begin with the tent map, and pick $\mu = 3$. For this value of μ it is known from 'symbolic dynamics' that there exists an uncountable (but measure zero) set of initial points x_0 for which the orbits do not leave the interval $0 \le x \le 1$; that there are periodic orbits of every period; and that the map is chaotic. Numerical simulation of this map using floating-point arithmetic does *not* confirm these results. In fact, I could not find *any* periodic points by using floating-point arithmetic. Instead, all the orbits 'escaped'—*i.e.* became bigger than 1 and thence went to infinity.

Interval arithmetic (which guarantees that the correct result of any floating-point operation is contained in the resulting interval [24]) is more reliable, but but still not illuminating; after a very short number of iterations, the interval containing the point in the orbit is just $[0, 1]$—the expansiveness of the map defeats the purpose of doing correct interval arithmetic. Let us try instead using exact arithmetic.

```
> T := proc(x:numeric)
>   if x < 1/2 then
>     3*x
>   else
>     3*(1-x)
>   fi
> end:
>
> T(1/10);

              3/10

> T(");
              9/10

> T(");
              3/10
```

So we have, relatively easily, found a periodic orbit. We now look at a *vector* of initial conditions to see if we can find any more.

```
> V := n -> vector(n, (i-> i)/(n+1) );
                              i -> i
      V := n -> vector(n, ------)
                               n + 1

> V(19);

  [ 1/20, 1/10, 3/20, 1/5, 1/4, 3/10, 7/20,


                         11          13
     2/5, 9/20, 1/2, ----, 3/5, ----,
                         20          20

                         17           19
     7/10, 3/4, 4/5, ----, 9/10, ---- ]
                         20           20

> map(T,");
                                            21
  [ 3/20, 3/10, 9/20, 3/5, 3/4, 9/10, ----,
                                            20

            27          27          21
     6/5, ----, 3/2, ----, 6/5, ----,
            20          20          20

     9/10, 3/4, 3/5, 9/20, 3/10, 3/20 ]
```

What we see is encouraging: we have found, with very little effort, one of the periodic orbits, and several true trajectories which end up on this periodic orbit. Further investigations uncover other periodic orbits, at very little cost.

Consider now the discrete logistic map, with $a = 3.6 = 36/10 = 18/5$. The results here are very different.

```
> h := 36/10;
                              h := 18/5

> L := t -> h*t*(1-t);
                        L := t -> h t (1 - t)
```

Because of floating-point calculations, we think that chaos occurs in this map for h near 3.6.

```
> x := 1/2;
                               x := 1/2

> nsteps := 14;
                             nsteps := 14

> times := array(1..nsteps):
> d_sizes := array(1..nsteps):
> for i to nsteps do
>   st := time(): x := L(x); times[i] := time()-st;
>   d_sizes[i] := ilog10(denom(x));
> od:
> print(times);
               [ 0, 0, 1.000, 0, 0, 0, 0, 0, 0, 0,
                 2.000, 6.000, 22.000, 91.000 ]

> print(d_sizes);
               [ 1, 2, 5, 10, 21, 44, 89, 178,
               357, 715, 1431, 2862, 5725, 11451 ]
```

The array `d_sizes` contains the number of digits in the denominator of the iterates. We see that the number of digits *more than doubles* with each iteration—that is, there is exponential growth in the size of the exact rational number used to represent the iterate. This leads to exponential growth in computing time, and indeed we see that the time to compute each iterate basically quadruples with each increment of n. Thus we can expect that it will take nearly 400 seconds to compute the next iteration, nearly 1600 for the one after that, and so on.

This shows that exact arithmetic is useless for this problem—at least, this naive attempt is useless.

We return now to the tent map, to see if we were just lucky with it the first time. If we change the 3 to 31/10, we see that indeed this was so—we now get exponential growth in the size of the rational numbers used to represent the iterate.

3 Elementary Use

However, if we are not so naive, we *can* use exact or 'arbitrary precision' floating-point arithmetic for (some) nonlinear maps. For example, consider exact computation of the first 8192 elements of the Gauss Map starting at $x = \pi - 3$ [12]. The Gauss map is defined as

$$G(x) = \begin{cases} 0 & \text{if } x = 0 \\ x^{-1} \bmod 1 & \text{otherwise} \end{cases}$$

and maps $[0, 1)$ to itself. This map is used in the computation of continued fractions; putting $n_0 = [x]$, the integer part of x, and $\gamma_0 = x - n_0$, the fractional part of x, and $\gamma_{k+1} = G(\gamma_k)$, then the integer parts of $1/\gamma_k$ are the entries n_k in the continued fraction expansion

$$x = n_0 + \cfrac{1}{n_1 + \cfrac{1}{n_2 + \cfrac{1}{\ddots}}} .$$

If we start with 8500 digits of π, then a theorem of Khintchin's [12, 19] says that we may *expect* that the first $8192 + 25$ partial quotients n_k of the computed continued fraction will be correct (this is easy to check afterwards). Then we can give *uniformly accurate* approximations to the orbit of G starting at $\pi - 3$ by using the fact that the Gauss map is just the 'shift map' on the continued fraction representation of γ_0: if $\gamma = [n_1, n_2, n_3, \ldots]$ then $G(\gamma) = [n_2, n_3, n_4, \ldots]$. Using some approximation theory of simple continued fractions, $\gamma_k = [n_k, n_{k+1}, \ldots, n_{k+25}] + \varepsilon$ where $|\varepsilon| < 10^{-6}$; this enables us to (relatively) cheaply follow the orbit of γ_0.

This example is not quite as *ad hoc* and special-purpose as it seems. What we have really used here is the 'Markov partition' representation of the orbit to give uniformly accurate floating-point approximations to the elements of the orbit.

Further, one can extend this idea to the case where we do not know *a priori* the initial condition to such accuracy—instead, we can compute *as we go along* the initial conditions that correspond to the computed orbit [22]. This topic will not be pursued here, except to note that this is the idea behind the strong shadowing result for the Gauss Map [11].

For an example of the use of the `networks` package in Maple to analyze the effectiveness of shadowing for this map, see [9].

4 Symbolic Algebra and the Logistic Map.

If, instead of a numerical value of a, we use a symbol in equation (1), it turns out that we can gain valuable algebraic insight into the bifurcations of the logistic map. We begin with the Maple syntax for the logistic map operator, which is the following.

```
> f := x -> a*x*(1-x);
```

$$f := x \to a\, x\,(\,1 - x\,)$$

This tells Maple that $f(u) = au(1-u)$, with a being a variable. In the following, the phrase **(f@@3)(x)** is the Maple syntax for $f^{(3)}(x) = f(f(f(x)))$.

```
> fn := (x,n) -> expand( (f@@n)(x) );
```

$$\mathrm{fn} \; := (x, n) \to \mathrm{expand}(f^{(n)}(x))$$

First, let us look at the fixed points of f, or the roots of $x - f(x)$.

```
> factor(x - f(x));
```

$$x\,(\,1 - a + a\,x\,)$$

The solutions are $x = 0$ and $x = 1 - 1/a$, which is positive if a is bigger than 1. We ignore the trivial solution $x = 0$.

```
> nontrivial := 1 - 1/a;
```

$$\mathrm{nontrivial} \; := 1 - \frac{1}{a}$$

These roots will be stable if the derivative is less than 1 in magnitude. Thus we investigate the locations where the derivative is equal to 1, and the locations where the derivative is equal to -1. We can simply evaluate $f'(x^*)$ where $x^* = 1 - 1/a$ is the nontrivial steady state. This evaluates to $2 - a$, and we can very simply see that $f'(x^*) = 1$ if $a = 1$ and $f'(x^*) = -1$ if $a = 3$, and thus $|f'(x^*)| < 1$ if $1 < a < 3$. This gives us all the stability information that we need. We now investigate the period-two points.

```
> f2 := fn(x,2);
```

$$\mathrm{f2} := a^2\,x - a^3\,x^2 + 2\,a^3\,x^3 - a^2\,x^2 - a^3\,x^4$$

```
> factor( x - f2 );
```

$$x\,(\,a\,x + 1 - a\,)\,(\,a^2\,x^2 - a^2\,x - a\,x + a + 1\,)$$

Notice that the period-1 points are also period-2 points. Only the last factor describes the true period-2 points.

```
> period_2 := op(3,");
```

$$\text{period\_2} := a^2\,x^2 - a^2\,x - a\,x + a + 1$$

We use Maple to find the *discriminant* of that polynomial. This algebraic quantity is very useful in this analytic context: the roots of the discriminant tell us which parameter values a may give rise to multiple roots in a polynomial. See [1] for an introduction to the theory of discriminants.

```
> factor(discrim(",x));
```

$$a^2\,(\,a+1\,)\,(\,a-3\,)$$

This discriminant will be zero precisely when there are multiple period-two points; the only positive root is at $a = 3$.

```
> solve(period_2,x):
```

```
> r1 := "[1]: r2 := ""[2]:
```

```
> r1 := map(factor,r1);
```

$$r_1 := \frac{1}{2}\,\frac{a^2 + a + \sqrt{a^2\,(\,a+1\,)\,(\,a-3\,)}}{a^2}$$

```
> r2 := map(factor,r2);
```

$$r_2 := \frac{1}{2}\,\frac{a^2 + a - \sqrt{a^2\,(\,a+1\,)\,(\,a-3\,)}}{a^2}$$

```
> plot({nontrivial,r1,r2},a=2..4,0..2);
```

A slightly modified version of this plot—including stability information—appears in Figure 1. We knew from before that the period-one solution loses stability at $(3, 2/3)$, which is precisely where the two new period-two points are 'born'. We now look at stability of these points. As before, we simply evaluate the derivative (of $f^{(2)}$ this time) at the period-2 points. It turns out that we can do this economically in Maple by handling both roots at once, using the `RootOf` construct, as follows.

```
> alias(alpha=RootOf(period_2,x));
```

$$I, \alpha$$

This tells Maple that α is *one of* the period-2 points. Maple has no way of telling which one, so it will apply only simplification rules valid for both, such as $a^2x^2 - a^2x - ax + a + 1 = 0$. We now evaluate the derivative of $f^{(2)}$ at $x = \alpha$.

```
> df2 := diff(f2,x):
```

```
> subs(x=alpha,");
```

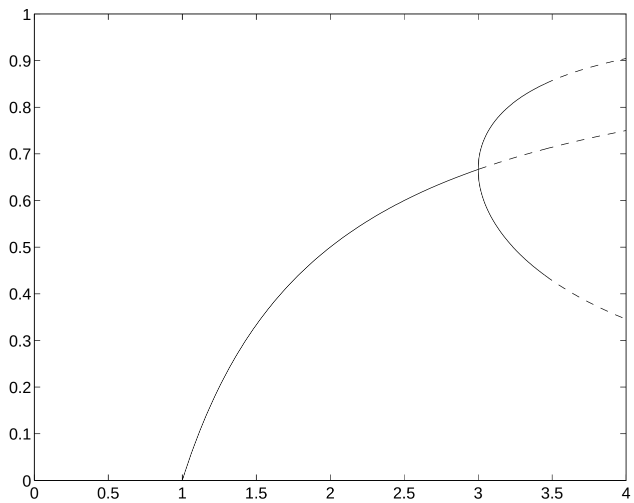

Figure 1: The nontrivial fixed-point and the period-2 points. The points are stable where the line is solid.

$$a^2 - 2\,\alpha\,a^3 + 6\,a^3\,\alpha^2 - 2\,a^2\,\alpha - 4\,a^3\,\alpha^3$$

That looks rather complicated, but we will see if Maple can simplify it a bit:

```
> simplify(");
```

$$-a^2 + 2\,a + 4$$

This is independent of α—that is, the value of the derivative depends only on a, not on which period-2 point we are on. This allows very simple determination of the stability.

```
> student[completesquare](",a);
```

$$-(\,a-1\,)^2 + 5$$

This is between 1 and -1 if and only if $3 < a < 1 + \sqrt{6} = 3.449\ldots$. Thus both period-2 points lose their stability at $a = 1 + \sqrt{6}$. To determine just how they lose their stability, we can investigate the period-4 points, where we see that there are multiple period-4 points at $a = 1+\sqrt{6}$ by a discriminant analysis. We do not continue this investigation here, for space reasons. Consult the worksheet `logmap.ms` in the Maple Share Library for a continuation of this analysis. The worksheet explores the period-4 and the period-8 points. For this map still more can be said, and indeed the use of symbolic manipulation can go quite far [4].

5 Derivatives

Computation of derivatives is probably the simplest yet most effective use of symbolic computation for nonlinear dynamics. This facility is much more useful for larger problems. Computation of Jacobian derivatives of vector functions by hand is tedious and error-prone, although a good human differentiator can still beat most computer algebra systems on special problems [16].

Computation of Taylor series or Lie series for use in a numerical method for solving differential equations is also extremely helpful in understanding the behaviour (e.g. for analysis of the singularity structure of the solution) of nonlinear initial-value problems.

6 Perturbation Expansions

Perturbation is one of the principal techniques by which nonlinear problems are attacked, and a comprehensive bibliography of computer-algebra implementations of perturbation techniques would be much longer than this paper itself. Hence I will confine myself to general remarks.

The computation of normal forms is a perturbation expansion that is particularly useful for bifurcation studies, as is the method of multiple scales [8]. These usually involve perturbation from linear, typically oscillatory, systems.

It is also possible to perturb from solvable nonlinear problems—in [5] a very good perturbation analysis of the forced Duffing equation was made by using elliptic functions.

The computation of Lyapunov exponents is essentially a perturbation calculation—just the first term, and hence only one (Jacobian) derivative need be taken, followed by the solution of a linear system of ordinary differential equations. Likewise Melnikov's method uses a clever perturbation scheme to arrive at an integral, the properties of which can tell us if the original system is chaotic. Computer algebra can sometimes help with the analysis of the integral.

7 A Problem in Flow-Induced Vibration

In [8], bifurcation of a mathematical model of flow-induced vibration was studied using computer algebra. The Macsyma program TWOVAR [25] was used to convert the differential equations to a simpler set by the method of multiple scales, and the (several parameter families of) equilibria of the resulting equations were found with Maple by use of Gröbner bases. The stability of these equilibria were studied in the usual fashion by finding the characteristic polynomial of the Jacobian matrix (which was also computed using Maple). The conditions for which the equilibria are stable were derived using the Hurwitz code from the Maple share library [10]. These conditions were then analyzed for bifurcations—that is, qualitative change in the nature or stability of the equilibria—by extensive use of polynomial factorization, discriminants, and graphics. Independent use of Matlab at selected points in the bifurcation diagram confirmed the symbolic results, which could not have been obtained by hand or numerically (the regions of qualitative change vary too widely in size to be completely found numerically).

One remarkable feature of this problem was the number of high-degree multivariate polynomials which happened to factor usefully. For example, the following discriminant was obtained in [8].

$$\begin{aligned}\Delta_{C_3} &= -\tfrac{1}{16} v^{10}\, s^6 d^6 \left(4\,v + v\,s^2 d^2 - s^2 d^2\right)^2 \\ &\qquad \times (4\,v - sd)\,(4\,v + sd)\left(1024\,v^4 - 27\,s^2 d^2\right) \ .\end{aligned}$$

The above factored form allowed simple and useful curves to be drawn in the bifurcation diagram in the sd—v plane, separating regions of qualitatively different behaviour. Ultimately, from all three stability conditions, nine different regions were identified in parameter space, giving a complete characterisation of the simple behaviour of the solutions to the flow-induced vibration model. The behaviour discovered included stable and unstable limit cycles, and regions in parameter space where two Hopf bifurcation points were joined by unstable limit cycles (this condition is often seen in period-doubling sequences leading to chaos).

8 The Method of Modified Equations

We give here a brief example of the method of modified equations [17], which is a technique used to examine the reliability of numerical methods for solving differential equations. This particular example is pursued further in [6] and [7].

In [3, *p.* 221] W.-J. Beyn gives the following didactic example to show that it is impossible in general to embed an arbitrary discrete dynamical system into a continuous one. Consider Euler's method, with fixed stepsize h, applied to the simple nonlinear problem $y' = y^2$. Then the resulting discrete dynamical system is $u \rightarrow u + hu^2$, which is not a diffeomorphism (the derivative is zero at $hu = -1/2$, and the inverse map is not unique), whereas the h-flow of any continuous dynamical system must be a diffeomorphism. Hence in some sense it is impossible to embed this discrete system in a continuous one. Beyn [3] then remarks that this proof relies on the global behaviour of the discrete flow, and conjectures that even locally this would be impossible (*i.e.* in some u-neighbourhood of 0).

We start our analysis here by computing a few terms in the h-series for the modified equation. As usual [17], we expand the local error of Euler's method applied to this problem in a Taylor series and set it to zero:

$$\begin{aligned}&\left(u(t+h) - u(t)\right)/h - u^2(t) = \\ &\qquad u'(t) - u^2(t) + \tfrac{1}{2!} h u''(t) + \tfrac{1}{3!} h^2 u'''(t) + \cdots\end{aligned}$$

Differentiating once to eliminate u'', and again to eliminate u''', and so on, we find that a fourth-order modified equation is

$$u' = (1 - hu + \tfrac{3}{2}(hu)^2 - \tfrac{8}{3}(hu)^3)u^2 \ .$$

This leads us to suspect a very simple form for the *infinite*-order modified equation that we wish to find, *viz*

$$u' = B(hu)u^2 \ . \tag{2}$$

We now simplify by nondimensionalizing. Put $v = hu$ and $\tau = t/h$, and then

$$\frac{dv}{d\tau} = B(v)v^2 \tag{3}$$

and

$$v(\tau + 1) = v(\tau) + v^2(\tau) \ . \tag{4}$$

A simple Maple program was written to compute more terms in the series for $B(v)$. Once a few more terms in the series were computed, the series was *recognized* [26, 27]. It turns out that this problem has already been solved, in [20], in the domain of formal power series.

The series for $B(v)$ can be constructed recursively as follows. If

$$B(v) = c_1 + c_2 v + c_3 v^2 + \cdots \tag{5}$$

(in a purely formal sense), then $c_1 = 1$ and

$$c_n = -\frac{1}{n-1} \sum_{i=1}^{n-1} \binom{n-i+1}{i+1} c_{n-i} \ , \quad \text{for } n > 1 \ , \tag{6}$$

where $\binom{n}{j}$ is the binomial coefficient [20]. This enables efficient calculation of any desired number of terms in the series for $B(v)$.

It is unclear just how much use the series (5) is at this point, so we use an alternate approach to get more information about $B(v)$. Differentiation of equation (4) and using (3) gives us the following functional equation for $B(v)$:

$$B(v) = \frac{(1+v)^2}{(1+2v)} B(v + v^2) \ . \tag{7}$$

This equation allows us to describe $B(v)$ completely, and, together with the series, to compute it efficiently and accurately.

We take $B(0) = 1$, as we expect from the series and from consistency of Euler's method as $h \to 0$. Now consider $B(-1/4)$, for example. If $B(-1/4)$

exists, then

$$\begin{aligned}
B(-1/4) &= \frac{9}{8}B(-3/16) \\
&= \left(\frac{9}{8}\right)\cdot\left(\frac{169}{160}\right)B(-39/256) \\
&= \cdots \\
&= \prod_{k=0}^{\infty}\frac{(1+v_k)^2}{(1+2v_k)} \qquad (8) \\
&= 1.4266762676859975\ldots
\end{aligned}$$

where $v_{k+1} = v_k + v_k^2$ with $v_0 = -1/4$ and the penultimate equality is tentative at this moment—pending proof that the product converges to a function which satisfies (7)—and where we have used the fact that $v_k \to 0^-$ as $k \to \infty$ if $v_0 = -1/4$ and hence assumed that $B(v_k) \to 1$.

Convergence of the infinite product in (8) is established by discovering that the asymptotic behaviour of v_k is $v_k \approx -1/(k - 1/v_0)$, for initial values v_0 in $(-1, 0)$, and hence the terms in the product are asymptotic to $1 + O(1/k^2)$ and so the product converges absolutely. Note that $1+2v_k \neq 0$ unless $v_0 = -1/2$.

Now consider v_0 near $-1/2$. Equation (7) gives

$$B(v_0) = \frac{(1+v_0)^2}{1+2v_0}B(v_0+v_0^2)\,,$$

and since $v_1 = v_0+v_0^2$ will be near $-1/4$ if v_0 is near $-1/2$ we see that $B(v)$ has a pole at $v = -1/2$. Note that this is exactly the place where the map $u \to u+hu^2$ fails to be diffeomorphic: $hu = v = -1/2$. This is not a coincidence.

We now consider pre-images of $-1/2$ under $v \to v + v^2$; these, too, will be poles (we cannot cancel out a pole with a zero unless $(1 + v_k) = 0$ which only happens if $v_k = -1$; but all the forward images of -1 are 0). We graph the first 2000 pre-images of the pole at $v = -1/2$ in Figure 2. These pre-images were computed using Matlab. This set of preimages approaches (and densely fills out) the Julia set of the quadratic map $v \to v + v^2$ since it approaches the α-limit set of the unstable fixed point $v = 0$ [13, p. 287]. One sees that there is an infinite number of poles of B, and as a consequence of the arbitrary approach of the Julia set of this map to the origin [13] we see that there are poles arbitrarily close to the point of expansion for the series 5. Thus the radius of convergence of the series 5 is zero. Further, there is a *natural boundary* preventing analytic continuation of the function defined by the infinite product 8 to the region outside the Julia set.

We are mainly interested in $B(v)$ for positive v, which is outside the Julia set. This means that $v_k \to \infty$ as $k \to \infty$. If we run the iteration (7) backwards, then we can get a convergent product. Define

$$u_{k+1} = \frac{2u_k}{1+\sqrt{1+4u_k}} \qquad (9)$$

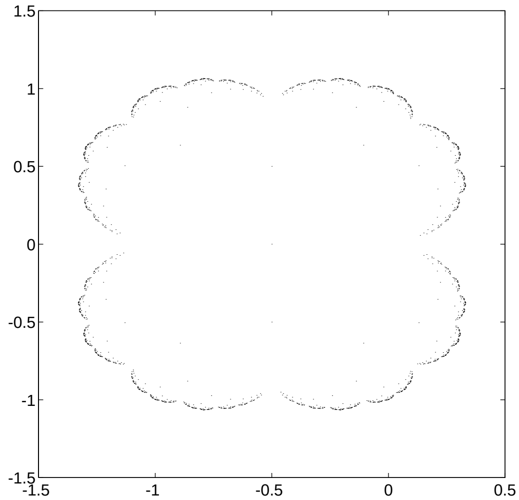

Figure 2: The first 2000 pre-images of $v = -1/2$ under the map $v \longrightarrow v + v^2$. These are locations of poles of $B(v)$, and as the number of pre-images increases, we see the poles approach the Julia set of the quadratic map $v \longrightarrow v + v^2$. In fact the poles are dense on that Julia set and form a natural boundary to analytic continuation of the function $B(v)$. The poles of $B(v)$ thus also come arbitrarily close to the origin.

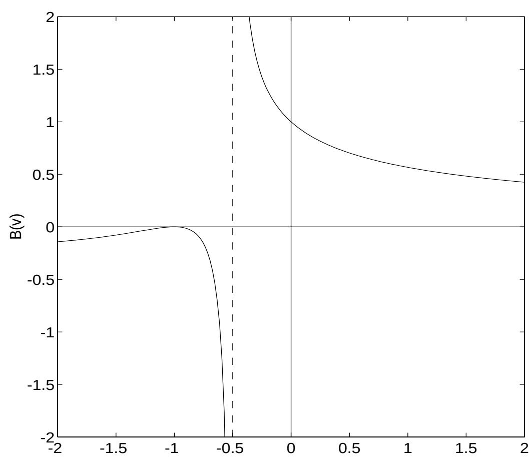

Figure 3: The graph of $B(v)$ for real v. The infinite-order modified equation is $y' = B(hy)y^2$, whose solution interpolates the Euler solution to $y' = y^2$ for initial conditions $y_0 \geq -1/(2h)$. $B(v)$ can be evaluated by use of two convergent infinite products, away from the only real pole at $v = -1/2$.

and, for $u_0 = v > 0$,

$$B(v) = \prod_{k=1}^{\infty} \frac{(1+2u_k)}{(1+u_k)^2} . \tag{10}$$

Note that the product starts at $k = 1$ this time, and that we have chosen a particular pre-image for each u_k by choosing u_{k+1} to be the root of $u+u^2 = u_k$ closest to zero. Note also that a numerically stable formula for this root has been used in (9). A similar analysis to that for the product (8) shows that this product converges for $v = u_0$ outside the Julia set, and by construction satisfies $B(0) = 1$ and the functional equation (7).

We can solve (3) up to quadrature by separation of variables (this provides a check for the product formulas for $B(v)$):

$$\int_{v_0}^{v_k} \frac{dv}{B(v)v^2} = \int_0^k dt = k . \tag{11}$$

It is an easy matter to verify by the change of variables $v = u + u^2$ (which can be done for real u and v so long as $v \geq -1/4$ and hence $u \geq -1/2$) that as a consequence of the functional identity (7),

$$\int_{v_{k-1}}^{v_k} \frac{dv}{B(v)v^2} = \int_{v_k}^{v_{k+1}} \frac{du}{B(u)u^2} = \text{constant} . \tag{12}$$

Using the asymptotics of v_k and of u_k as $k \to \infty$ we can show this constant is 1. This is easily confirmed by numerical quadrature in Maple.

To evaluate $B(v)$ in practice it turns out that the divergent series (5) is useful. We can use the Maple `evalf/Sum` program, which uses Levin's u-transform [28] for convergence acceleration; as well, this method will sum certain divergent series. For this example, Levin's u-transform is successful for real v in $[-0.1, 0.1]$ (for settings of `Digits` ≤ 30) and this is precisely the region where convergence of the infinite products (8) and (10) is slowest.

We have thus found a differential equation (2) whose solution interpolates the Euler's method solution of the original problem, for initial conditions $v_0 \geq -1/2$. The singularity at $v = -1/2$ shows that this is not a dynamical system in the ordinary sense, and thus supports Beyn's original observation [3].

More importantly, the differential equation (2) is a *large* perturbation of $y' = y^2$, *no matter how small* h *is*, once y gets large enough. However, on compact y-sets, we see that by taking h small enough, we can get the exact solution of a differential equation arbitrarily close to the original problem.

Generalizations to this are pursued in [7]. In particular, solution of Hamiltonian problems by symplectic methods is discussed, and Maple is used to generate good approximations to the Hamiltonians of the 'nearby' problems that the numerical methods actually solve.

9 Concluding Remarks

Symbolic computation can be profitably used in the study of nonlinear dynamical systems, although one must be careful not to expect too much of exact arithmetic. The facilities for polynomial manipulation, factoring, taking discriminants, and rootfinding are particularly useful. One can use Maple as a numerical laboratory to examine the effects of floating-point arithmetic, and one can use series expansions and modified equations to help to understand the effects of discretization and the reliability of numerical methods.

Acknowledgements

This work was supported by NSERC. Bruno Salvy pointed out the reference [20]. I would like also to thank the students at the 1993 International Summer School *Let's Face Chaos through Nonlinear Dynamics* at the University of Ljubljana, and my Applied Computer Algebra graduate students.

References

[1] E. J. Barbeau, *Polynomials*, Springer-Verlag Problem Books in Mathematics, 1989.

[2] H. H. Bau, T. Herbert, and M. M. Yovanovich, eds, *Symbolic Computation in Fluid Mechanics and Heat Transfer*, ASME HTD-vol 105 AMD-vol 97, 1988.

[3] Wolf-Jürgen Beyn, "Numerical Methods for Dynamical Systems" in *Advances in Numerical Analysis*, **I**, Will Light, ed., Oxford Science Publications, 1991, *pp.* 175–236.

[4] Keith Briggs, *personal communication*

[5] Vincent T. Coppola and Richard H. Rand, "Symbolic Computation and Perturbation Methods Using Elliptic Functions", Trans. Sixth Army Conference on Applied Mathematics and Computing, 1989, *pp.* 639–676.

[6] Robert M. Corless, *Symbolic Recipes, vol II*, Springer-Verlag *to appear* 1994.

[7] Robert M. Corless, "Error Backward", in *Proceedings of Chaotic Numerics*, Geelong, 1993, Peter E. Kloeden and Ken J. Palmer, eds. *to appear*.

[8] Robert M. Corless, "Bifurcation in a flow-induced vibration model", *to appear* in Proc. Fields Institute Workshop on Normal Forms and Homoclinic Chaos, W. Nagata and W. Langford, eds.

[9] Robert M. Corless, "What good are numerical solutions of chaotic differential equations?", *to appear in Computers and Mathematics with Applications.*

[10] Robert M. Corless, "HURWITZ", *Maple share library, 1990.*

[11] Robert M. Corless, "Continued Fractions and Chaos", American Mathematical Monthly, **99** no. 3, March 1992, *pp.* 203–215.

[12] Robert M. Corless, Gregory W. Frank, and J. Graham Monroe, "Chaos and Continued Fractions", Physica D, **46** 1990, *pp.* 241–253.

[13] Robert L. Devaney, *An Introduction to Chaotic Dynamical Systems*, 2nd. ed., Addison-Wesley, 1989.

[14] Walter Gander and Jiři Hřebiíček, *Solving Problems in Scientific Computing using Maple and MATLAB*, Springer-Verlag, 1993.

[15] D. Gavelek & T. Erber, 1992, "Shadowing and Iterative Interpolation for Čebyšev Mixing Transformations", J. Comp. Phys. **101** *pp.* 25–50.

[16] E. Katende, A. Jutan, and Robert M. Corless, 'A problem in automatic control', *to appear.*

[17] D. F. Griffiths and J. M. Sanz-Serna, "On the scope of the Method of Modified Equations", SIAM J. Sci. Stat. Comput., **7** , No. 3, 1986, *pp.* 994–1008.

[18] E. Kaltofen and S. M. Watt, eds, *Computers and Mathematics*, Springer-Verlag, 1989.

[19] A. Y. Khintchin, *Continued Fractions*, P. Noordhoff (Gröningen) 1963.

[20] G. Labelle, "Sur l'Inversion et l'Itération Continue des Séries Formelles", Europ. J. Combinatorics, **1** , 1980, *pp.* 113–138.

[21] Thomas Lee, ed., *Mathematical Computation with Maple V: Ideas and Applications*, Birkhäuser, 1993.

[22] J. L. McCauley, *Chaos, Dynamics, and Fractals*, Cambridge, 1993.

[23] Kenneth R. Meyer and Dieter S. Schmidt, eds, *Computer Aided Proofs in Analysis*, IMA vol. 28, Springer-Verlag, 1991.

[24] Ramon E. Moore, *Methods and Applications of Interval Analysis*, SIAM, Philadelphia, 1979.

[25] Richard H. Rand and Dieter Armbruster, *Perturbation Methods, Bifurcation Theory, and Computer Algebra*, **65** Applied Mathematical Sciences, Springer-Verlag , New York, 1987.

[26] Bruno Salvy, *personal communication*

[27] I. Sloane, *Handbook of Integer Sequences*, 2nd ed.

[28] Ernst Joachim Weniger, *Nonlinear Sequence Transformations for the Acceleration of Convergence and the Summation of Divergent Series*, Computer Physics Reports **10**, North-Holland, 1989.