Computing minimal polynomials for bifurcation points of the logistic map.

Michael Monagan,
Department of Mathematics, Simon Fraser University.
mmonagan@cecm.sfu.ca

Extended Abstract

Recall that logistic map is the function

\[ f(x) = ax(1-x) \]

with parameter \( a \). Consider applying \( f(x) \) to a value \( x_0 \in (0, 1) \) to generate the sequence \( x_1, x_2, x_3, \ldots \) where

\[ x_1 = f(x_0), \; x_2 = f(f(x_0)), \ldots, \; x_k = f(x_{k-1}) = f^k(x_0). \]

It is known that this sequence converges to a one-cycle for \( 1 < a < 3 \). For \( 3 < a < 1 + \sqrt{6} \), the sequence converges to a two-cycle, and beyond this there is a stable 4-cycle. Thus there are bifurcations at \( a = 3 \) and \( a = 1 + \sqrt{6} \). Beyond \( a = 1 + \sqrt{6} \), a period-doubling bifurcation sequence occurs, that is, we find a stable 4-cycle, then 8-cycle, 16-cycle, etc.

Let \( B_n \) denote the bifurcation point between the stable cycles of periods \( n \) and \( 2n \). Many numerical methods (we will describe one in the talk that uses automatic differentiation) have been developed to compute the \( B_n \). It turns out that the \( B_n \) are algebraic numbers and so we may speak of their minimal polynomials \( M_n(a) \). The first few bifurcation points and their minimal polynomials are given in the table below.

<table>
<thead>
<tr>
<th>( n )</th>
<th>( B_n )</th>
<th>( M_n(a) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>( a - 3 )</td>
</tr>
<tr>
<td>2</td>
<td>1 + \sqrt{6}</td>
<td>( a^2 - 2a - 5 )</td>
</tr>
<tr>
<td>4</td>
<td>3.498561699</td>
<td>( a^{12} - 12a^{11} + 48a^{10} - 40a^9 - 192a^8 + 384a^7 + 64a^6 - 1024a^4 - 512a^3 + 2048a^2 + 4096 )</td>
</tr>
</tbody>
</table>
The next bifurcation point is $B_8 = 3.564407266$. Computing $M_8(a)$ is not easy. The polynomial $M_8(a)$ has degree 240 and 73 digit integer coefficients. Computing $M_{16}(a)$ is MUCH harder. The polynomial $M_{16}(a)$ has degree just under $2^{16}$ and its integer coefficients have length $2^{16}$ bits. That is, the total size of $M_{16}(a)$ is $2^{32}$ bits or about half a Gigabyte. We’d like to set computing $M_{16}(a)$ as a computational challenge. It may well be that computing $M_{16}(a)$ is just not possible. As steps towards this challenge, we propose to compute the minimal polynomials $M_9(a)$, $M_{10}(a)$, $M_{11}(a)$, ... for the bifurcation points between the $n$-cycles and $2n$-cycles for $n = 9, 10, 11, ..., 15$ and, finally $n = 16$. Since the degree of $M_n(a)$ is approximately $2^n$ and the size of its coefficients are approximately $2^n$ bits, increasing $n$ by 1 quadruples the size of $M(a)$.

In this talk we consider three methods for computing $M_n(a)$. The first is described by Bailey et. al. in their paper “Ten Problems in Experimental Mathematics” – see [1]. In outline, one first approximates $B_n$ to high precision using a numerical method. Next, assuming that the degree of $M_n(a)$ is known to be less than $N$, one applies Ferguson’s PSLQ algorithm to search for a integer relation between the decimal numbers $1, B_n, B_{2n}, ..., B_{Nn}$. This gives the coefficients of $M_n(a)$. Bailey et. al. used this method to determine $M_8(a)$. It required that $B_8$ be computed to over 10,000 digits of precision. The numerical precision needed for $B_n$ is a little more than $\deg(M_n) \log_{10} ||M_n||_\infty$ decimal digits, that is, about the size of $M_n(a)$.

In [2], Kotsirias and Karamanos describe a method for computing $M_n(a)$ which is purely algebraic. It does not compute $B_n$, but rather uses Groebner bases to do an elimination to compute $M_n(a)$. We illustrate the method for $n = 2$. The 2-cycle of the logistic map can be defined by the equations.

$$
x_2 = ax_1(1 - x_1), \ x_3 = ax_2(1 - x_2) \text{ and } x_3 = x_1.
$$

The bifurcation occurs when the stability of the map is $\pm 1$. This occurs when $[f(f(x))]' = +1$. From this one obtains $a^2(1 - 2x_1)(1 - 2x_2) = -1$. One constructs the ideal

$$I = \langle x_2 - ax_1(1 - x_1), x_1 - ax_2(1 - x_2), a^2(1 - 2x_1)(1 - 2x_2) + 1 \rangle$$

in $\mathbb{Q}[a, x_1, x_2]$ and computes generators for

$$I \cap \mathbb{Q}[a]$$

using Groebner bases. Since $I \cap \mathbb{Q}[a]$ is a principal ideal, $M_2(a)$ is a factor of a generator of $I \cap \mathbb{Q}[a]$. The authors reported in [2] that it took 5 and a half hours in Magma to compute $M_8(a)$ using this method.
We will present a third method that is semi-numerical. In principle, $M_n(a)$ can be found using resultants to eliminate $x$ from a factor of $f^{(n)}(x) - x$ and the polynomial $[f^{(n)}(x)]_1^n + 1$. For $n = 2$ one obtains the polynomial

$$a^6 \left(a^2 - 2a - 5\right)^2$$

which is not equal to $M_2(a)$. In general the resultant we obtain is of the form $a^{L_n} M_n(a)^{D_n}$ where $L_n$ is approximately $2^{2n}$ and $D_n = n$. The very large factor $a^{L_n}$ is a problem for the modular resultant algorithm of Collins which interpolates $M_n(a)$ modulo a sequence of primes. In [3], we modified Collins’ modular resultant algorithm to automatically detects the high low degree $L_n$ in such a way that the number of interpolation points used is $O(D_n \deg M_n(a))$ instead of $O(L_n D_n \deg M_n(a))$.

In current work we have refined the method so that it reconstructs the square-free part of the resultant to save a factor of $D_n$. We also now have an exact formula for $L_n$ which avoids the need to bound $L_n$. Using our new algorithm we can compute $M_8(a)$ in under 2 minutes on a desktop computer – a single core AMD Opteron running at 2.4 GHz. To compute $M_n(a)$, the new algorithm is $O(N^2 \log N)$ where $N = 2^{2n}$ is the size of $M_n(a)$. Thus it takes approximately 16 times longer to compute $M_{n+1}(a)$ than $M_n(a)$. Hence, we estimate that it would take $2 \times 16^8 = 8,589,934,592$ minutes to compute $M_{16}(a)$ – which is clearly not feasible.

In the talk I will describe in more detail our modular algorithm. The algorithm is embarrassingly parallel so this is one way to try to speed it up. I will present the results we have computed so far for the sequence of minimal polynomials $M_9(a)$, $M_{10}(a)$, $M_{11}(a)$, … . One of the interesting aspects of the algorithm is that it uses automatic differentiation.

References

