Computation of Liapunov exponents for maps

Marc R. Roussel

We will use the logistic equation as an example:

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

$$x \to \lambda\, x\, (1 - x) \tag{1}$$

We can use Maple to calculate the derivative. The following technique makes the derivative into a function, which is particularly convenient:

```
> dfdx := unapply(diff(f(x),x),x);
```

$$x \to \lambda\, (1 - x) - \lambda\, x \tag{2}$$

We're going to set up a calculation for a few hundred different values of l. The interesting region is from l=3 and up. We will use 1000 iterates to calculate each Liapunov exponent. Each trajectory will start from a non-rational value.

```
> npts := 300;
```

$$300 \tag{3}$$

```
> lambda_min := 3;
```

$$3 \tag{4}$$

```
> lambda_max := 4;
```

$$4 \tag{5}$$

```
> niter := 1000;
```

$$1000 \tag{6}$$

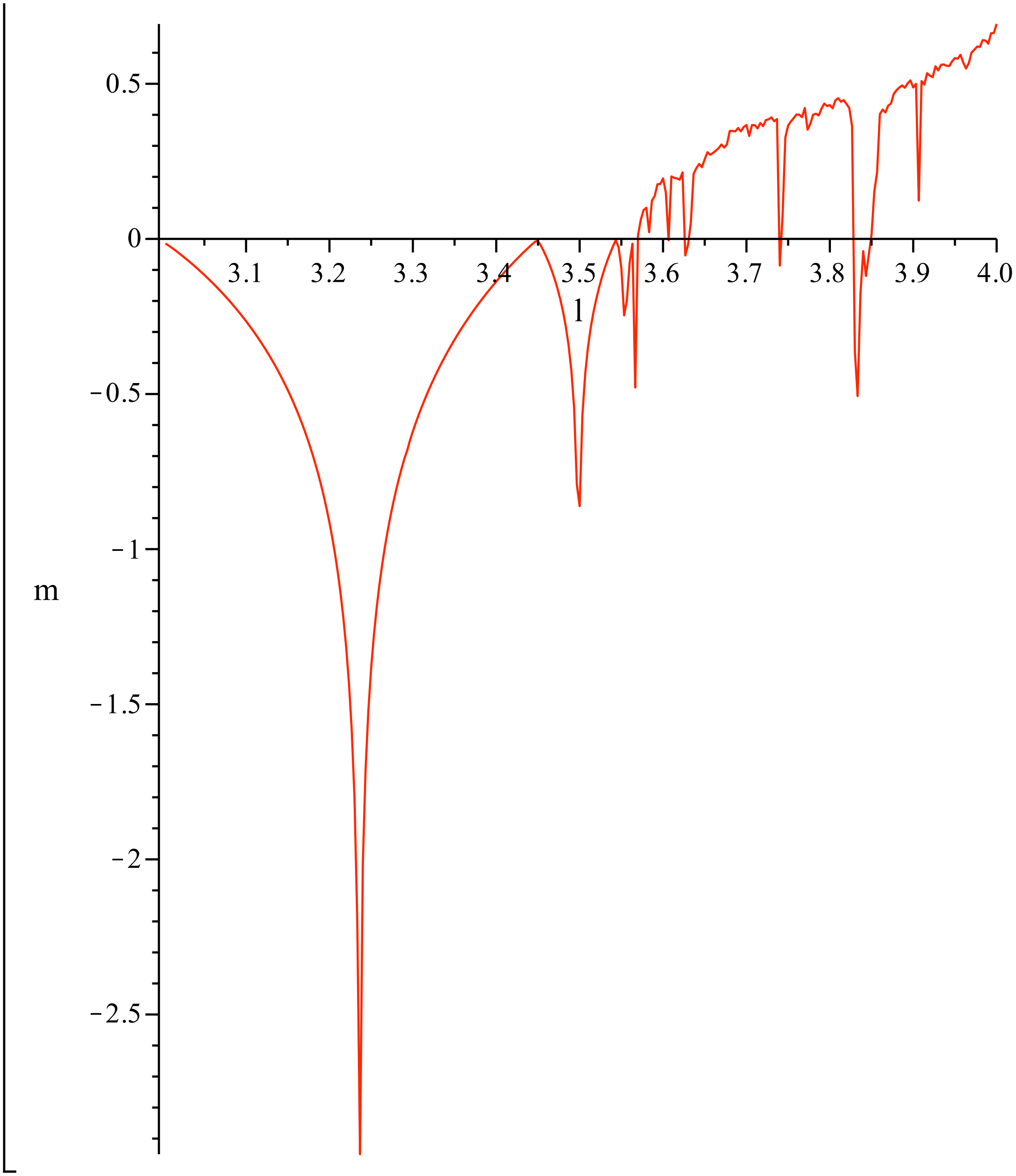
```
> x0 := evalf(Pi/4);
```

$$0.7853981635 \tag{7}$$

The following loop does all the work. The outer loop increments the value of l. The variable liapexp is used as an accumulator which must be reset for each new value of l. The inner loop computes iterates of the map and the sum required for the Liapunov exponent. The rest is simple bookkeeping.

```
> for i from 1 to npts do
> lambda := evalf(lambda_min + (i/npts)*(lambda_max-lambda_min));
> liapexp := 0;
> x := x0;
> for j from 1 to niter do
> x := f(x);
> liapexp := liapexp + ln(abs(dfdx(x)));
> od;
> liapexp := liapexp/niter;
> if i=1 then ptlist := [lambda,liapexp];
> else ptlist := ptlist,[lambda,liapexp]; fi;
> od:
```

```
> plot([ptlist],labels=["l","m"],labelfont=[SYMBOL]);
```


0.5
0
−0.5
−1
−1.5
−2
−2.5
m
3.1
3.2
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4.0
l