3. Iterations

A function consists of a set of instructions that convert given input values to output values. Now if the output itself is part of the domain of the function considered, it can become an input in turn, and the function returns new output values. This process can then be repeated indefinitely. While there are often no analytic methods to calculate the structural properties of such iterations of the form $x_{t+1} = f(x_t)$, they are easily generated on the computer. Obviously, one just has to apply the same function $f(x)$ to the result repeatedly. We want to demonstrate this with a few examples.

3.1 Population Dynamics

Arguably, the best-known iteration of a nonlinear function is the so-called *logistic map*. It is a simple parabola that projects the real numbers of the unit interval onto themselves. We want to present it here because every scientist should know the properties of such an elementary iteration.

A simple mechanism leads to complex behavior – this is the result of the repeated application of a quadratic function. Although there are only a few analytic results on this, anyone can easily reproduce the iteration with a pocket calculator. For certain parameter values of the parabola the iterated numbers jump around in the unit interval in an irregular manner. The sequence of numbers is extremely sensitive to changes of the starting value of the iteration. A well-defined function definition yields a sequence of numbers that is practically unpredictable even for a small initial uncertainty. Such a behavior is called *deterministic chaos.*

Some quantities that can be calculated with the computer are not just the result of a mathematical folly; rather, one finds these numbers in many experiments near the transition from regular to chaotic behavior. Such a universality of quantitative values is an additional fascinating aspect of this equation.

Physics

We consider the iteration

$$x_{n+1} = 4rx_n(1-x_n) = f(x_n) \ , \quad n = 0,1,2,3,\dots \ , \tag{3.1}$$

which was first introduced in 1845 by the Belgian biomathematician P.F. Verhulst, in a work on population dynamics. Here, x_n is iterated in the interval $[0,1]$, and r is a parameter that can be varied in the range $[0,1]$ as well.

The function (3.1) is of interest for various questions:

1. As a simple model for the time development of a growing population. In this model, x_n is proportional to the population density of a species at time n. Its growth is described by a linear contribution $4rx_n$ which, taken by itself, leads to exponential growth, i.e., a population explosion, for $r > 1/4$. The population growth is limited, however, by the nonlinear contribution $-4rx_n^2$ (due, e.g., to a limited food supply).
2. As a simple example of classical mechanics with friction. We will see in Sect. 4.2 that the motion of a mechanical system can be analyzed by a Poincaré section in phase space. This map results in an iteration of a nonlinear function. Therefore, it is useful to know what properties such iterations already have in one dimension, even though one-dimensional Poincaré sections in mechanics cannot exhibit chaos.
3. As an example of a (bad) numerical solution of a differential equation. If neighboring x_n values differ only by small amounts, then n and x_n can be extended to real numbers in a continuous way. If we set

   $$x_{n+1} - x_n \simeq \frac{\mathrm{d}x}{\mathrm{d}n} \ ,$$

   then (3.1) becomes an approximation of the differential equation

   $$\frac{\mathrm{d}x}{\mathrm{d}n} = (4r-1)\,x - 4rx^2$$

   with the solution

   $$x(n) = \frac{4r-1}{4r + \text{const}\cdot\exp((1-4r)\,n)} \ .$$

   For $r < 1/4$ this equation has the attractor $x_\infty = 0$, whereas for $r > 1/4$ all initial values $x(0) > 0$ lead to the value $x_\infty = 1 - 1/(4r)$. Like the discrete iteration, this equation has a phase transition at $r_0 = 1/4$ from a vanishing population to a constant one. It will turn out, however, that for larger values of r, the differential equation no longer bears any resemblance to its discrete approximation.

For $r > r_0 = 1/4$, (3.1) has a fixed point $x^* = f(x^*)$ with $x^* = 1 - 1/(4r)$. We now consider a small deviation ε_0 from the fixed point and iterate f for $x^* + \varepsilon_0$. Because of the relation $f(x^*+\varepsilon_0) \simeq f(x^*) + f'(x^*)\varepsilon_0 = x^* + f'(x^*)\varepsilon_0$ we have for small ε_n:

$$\varepsilon_n \simeq f'(x^*)\,\varepsilon_{n-1} \simeq [f'(x^*)]^n\,\varepsilon_0 . \tag{3.2}$$

Therefore, the perturbation ε_0 explodes for $|f'(x^*)| > 1$, which corresponds to $r > r_1 = 3/4$. We will see that, in this case, the x_n jump back and forth between the values x_1^* and x_2^* after a few steps:

$$\begin{aligned} &x_1^* = f(x_2^*)\ ,\ x_2^* = f(x_1^*)\ , \\ &\text{so}\ \ x_i^* = f(f(x_i^*))\ \ \text{for } i = 1,2 . \end{aligned} \tag{3.3}$$

This means that for certain values of r, the iterated function $f^{(2)}(x) = f(f(x))$ has two stable fixed points.

Now $|f^{(2)\prime}(x_i^*)| > 1$ for $r > r_2 = (1+\sqrt{6})/4$. Therefore, the same argument as above shows that the cycle of length 2 (or two-cycle, for short) becomes unstable for $r > r_2$; the x_n run into a four-cycle. This doubling of periods continues until their number reaches infinity at $r_\infty \simeq 0.89$.

A doubling of the period means reducing the frequency by half. Consequently, subharmonics in the frequency spectrum are generated. At r_l, the 2^{l-1}-cycle becomes unstable, resulting in a 2^l-cycle. As $l \to \infty$, the r_l values get closer and closer to r_∞, according to the following pattern:

$$r_l \approx r_\infty - \frac{c}{\delta^l} . \tag{3.4}$$

From this we can conclude

$$\delta = \lim_{l\to\infty} \frac{r_l - r_{l-1}}{r_{l+1} - r_l} . \tag{3.5}$$

The number δ is called the *Feigenbaum constant* and has the value $\delta = 4.6692\ldots$. It receives its importance from the fact that it is a universal quantity, i.e., it is observed for many different functions $f(x)$. Even experiments with real nonlinear systems that undergo a transition to chaos via period doubling obtain the same number.

For larger values of r, one obtains chaotic bands: x_n appears to jump around randomly, without a period, in one or more intervals. This holds true even though the x_n values are not generated randomly, but by the well-defined parabola $f(x)$ from (3.1). As mentioned before, such a behavior is called deterministic chaos.

In the chaotic region, one can use the simple parabola to study an essential property of deterministic chaos, the sensitivity to the initial conditions. Two extremely close values separate after just a few iterations and jump around in the chaotic bands seemingly independently. For a quantitative characterization, we designate the two orbits by x_n and $x_n + \varepsilon_n$. Then, for $\varepsilon_0 \to 0$

$$\varepsilon_n \simeq \varepsilon_0 \mathrm{e}^{\lambda n} . \tag{3.6}$$

The parameter λ is called the Lyapunov exponent; it is a measure of the sensitivity to the initial state. A positive exponent $\lambda > 0$ indicates chaos. We have

$$\left| f^{(n)}(x_0 + \varepsilon_0) - f^{(n)}(x_0) \right| \simeq \varepsilon_0 \mathrm{e}^{\lambda n} . \tag{3.7}$$

In the limit $\varepsilon_0 \to 0$ this yields the derivative of the n-fold iterated function $f^{(n)}(x_0) = f(f(\cdots f(x_0))\cdots)$. With the compound rule of three one immediately gets

$$\lambda = \lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \ln |f'(x_i)| \ . \tag{3.8}$$

Thus, in order to determine λ, only one orbit $\{x_i\}$ needs to be calculated and the logarithms of the slope of f at the points x_i have to be added up.

Algorithm

With *Mathematica* one can obtain an n-fold iteration of a given function, such as `f[x_] = 4 r x (1-x)`, simply by using

```
iterf[n_]:= Nest[f,x,n]
```

The determination of the bifurcation points r_l and consequently the Feigenbaum constant δ is significantly more difficult, however. One can find r_1 and r_2 by solving the two equations, $f^{(l)}(x^*) = x^*$ and $|\mathrm{d}f^{(l)}(x^*)/\mathrm{d}x| = 1$, for the unknown variables x^* and r_l, either by hand or by using the *Mathematica* function `FindRoot`. But we were unable to determine higher r_l values this way. It is sufficient, however, to consider so-called superstable periodic orbits. These are orbits that start at $x_0 = 1/2$ and return there after 2^l steps. Because $f'(1/2) = 0$, deviations from x_0 converge towards the 2^l-cycle very quickly. In each interval $[r_l, r_{l+1}]$ there is one superstable orbit with a period 2^l, at a value R_l. Here, the R_l values, like the r_l values, scale with the Feigenbaum constant δ as they approach r_∞ (see (3.5)).

Now again, solving the equation

$$f^{(2^l)}\left(\frac{1}{2}\right) = \frac{1}{2}$$

numerically is not easy. Since this equation has $2^l - 1$ additional fixed points, the function oscillates very rapidly and `FindRoot` fails. But there is a trick: one inverts the equation above. Since the inverse of f has two branches, one has to specify in the iteration whether x_n lies to the right (R) or left (L) of $x_0 = 1/2$. Therefore, any periodic orbit is determined by a sequence of C, R and L, where C (=center) denotes the initial value $x_0 = 1/2$ and R, L indicate whether x_i lies to the right or left of the maximum. Thus, the iteration is described by *symbolic dynamics* subject to certain rules.

There is a simple algorithm for determining the sequence of Rs and Ls for a given period 2^l. We describe it without proof. The initial value is supposed to be $x_0 = 1/2$, i.e., the sequence always starts with a C. For $l = 1$, we obviously get CR. The sequence for $l + 1$ is constructed from the one for l by doubling the l-sequence and replacing the C in the middle by L if the number of Rs in the l-sequence is odd, or by R if it is even. For $l = 2$, the doubling yields *CRCR* and this becomes *CRLR*, as *CR* contains just one *R*. For $l = 3$, one gets *CRLRRRLR*.

The inverse iteration is most easily performed in a coordinate system whose origin is at $(x, y) = (1/2, 1/2)$ and whose two axes are scaled by a common factor such that the peak of the parabola in the new system is at $(0, 1)$. In this coordinate system, the original parabola $f(x) = 4rx(1-x)$ takes the form

$$g(\xi) = 1 - \mu\xi^2 ,$$

where μ and r are related via

$$\mu = r(4r-2) \quad \text{or} \quad r = \frac{1}{4}\left(1 + \sqrt{1+4\mu}\right) .$$

We designate the left branch of the parabola by $g_{\mathrm{L}}(\xi)$ and the right one by $g_{\mathrm{R}}(\xi)$. Then the superstable orbit for $l = 2$ obeys the equation

$$g_{\mathrm{R}}(g_{\mathrm{L}}(g_{\mathrm{R}}(1))) = 0$$

or

$$1 = g_{\mathrm{R}}^{-1}(g_{\mathrm{L}}^{-1}(g_{\mathrm{R}}^{-1}(0)))$$

with

$$g_{\mathrm{R}}^{-1}(\eta) = \sqrt{\frac{1-\eta}{\mu}} \quad \text{and} \quad g_{\mathrm{L}}^{-1}(\eta) = -\sqrt{\frac{1-\eta}{\mu}} .$$

This finally leads to the following equation for μ:

$$\mu = \sqrt{\mu + \sqrt{\mu - \sqrt{\mu}}} , \tag{3.9}$$

which can be solved by iteration or by using `FindRoot`. The sequence *CRLR* determines the signs of the nested roots in (3.9). After the initial *CR*, which does not enter the equation above, *L* yields a positive and *R* a negative sign. This holds for all superstable periodic orbits. Thus, the orbit of period 5, with the sequence *CRLRR*, which is located between chaotic bands, is found at a μ value that obeys the following equation:

$$\mu = \sqrt{\mu + \sqrt{\mu - \sqrt{\mu - \sqrt{\mu}}}} .$$

We have programmed the generation of the sequence in *Mathematica* in the following way: For each symbol *R* (*L*), we write the numbers 1 (0); the sequence of length 2^n is stored as a list `period[n]`. Initially we define the list `period[1]= {c,1}` for $n = 1$, and the doubling of the sequence is done by

```
period[n_]:= period[n]=
    Join[period[n-1], correct[period[n-1]]]
```

Here, the symbol c still has to be replaced by 0, if the frequency with which the number 1 appears (= sum of the array elements) is odd, or by 1 if the sum is even.

```
correct[list_]:=Block[{sum=0,li=list,l=Length[list]},
                      Do[sum+=li[[i]],{i,2,l}];
                      If[OddQ[sum],li[[1]]=0,li[[1]]=1];
                      li]
```

The multiply nested root from the equation for μ is obtained by

```
g[n_,mu_]:=Block[{x=Sqrt[mu],l=Length[period[n]]},
    Do[x=Sqrt[mu+(-1)^(period[n][[i]]) x],{i,l,3,-1}];
             x]
```

Finally, this equation is solved numerically, which requires increasing the accuracy with which FindRoot calculates the solution:

```
prec=30
maxit=30
fr[n_]:=fr[n]=(find=FindRoot[g[n,mu]==mu,
                       {mu,{15/10,16/10}},
                       AccuracyGoal->prec,
                       WorkingPrecision->prec,
                       MaxIterations->maxit];
                       mu/.find)
```

In order to generate and display the orbits $x_0, x_1, x_2, x_3, \ldots$ for all parameters r (or μ), one should use a fast programming language, if possible, since each value of r requires about 1000 iterations. In principle, however, this is very simple: For the r value, one picks, for example, the x-coordinate of the pixels on the screen. For each value of r, one first calculates about 100 values of x_n, until the x_n have come very close to the attractor. Then, one calculates 1000 additional values of x_n, transforms them into the y-coordinate of the pixels, and plots a point at each pixel (x, y) obtained in this way. In C, this is written as:

```
xit = 4.*r*xit*(1.-xit);
putpixel( r*MAXX, (1.-xit)*MAXY,WHITE);
```

The parameter r is incremented or decremented one step at a time in response to keyboard input.

The density of the x_n values is of interest as well. To display it, the x-axis is subdivided into small intervals, e.g., of the order of the distance between adjacent pixels. Each interval is associated with a y-coordinate, which is initially set to the value $y = 0$. In the iteration of the parabola, for a fixed value of r, each x_n value falls into one of these intervals, whereupon the corresponding y-coordinate is incremented by one pixel and a point is plotted at the resulting (x, y) pixel. The result is a histogram of the x_n values which corresponds to the density of points on the attractor. This is done with the following function:

```
void channel( double r)
{
    double xit=.5;
    int x,y[600],i;
    clearviewport();
    for (i=0;i<MAXX;i++) y[i]=0;
    rectangle(0,0,MAXX,300);
    while(!kbhit())
    {    xit=4.*r*xit*(1.-xit);
          x=xit*MAXX;
         y[x]++;
         putpixel(x,300-y[x],WHITE);
     }
      getch();getch();clearviewport();return;
}
```

Results

Figure 3.1 shows the iteration starting from $x_0 = 0.65$ for $r = 0.87$. The x_n values move toward an attractor of period 4, which is described by the fixed points of the fourfold iterated function $f(x)$. These can be seen in Fig. 3.2. The function $f^{(4)}(x)$ intersects the straight line $y = x$ eight times, but only four of those are stable fixed points. As r is increased, attractors with the periods $8, 16, 32, \ldots$ are generated until the transition to chaos occurs at $r_\infty = 0.89\ldots$. This is clearly visible in Figs. 3.3 and 3.4, in which 1000 x_n values are plotted for each value of r, after an initial transition period of about 100 steps. Thus, attractors with a period p are visible as p points, whereas chaotic orbits fill several intervals on the vertical. Since at most 1000 points were plotted for each value of r, one also gets an impression of the density of x_n values. This, however, is more clearly visible in the histogram in Fig. 3.5. Periodic orbits with period p show up as a series of lines with p peaks, whereas chaotic orbits jump back and forth between various bands, each with continuous densities. Figure 3.5 shows the chaotic motion just

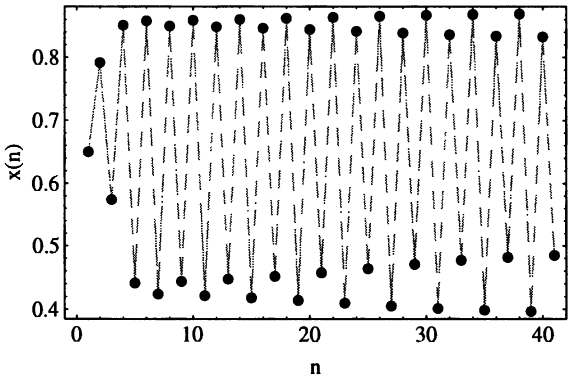

Fig. 3.1. Iteration starting from $x_0 = 0.65$ using the parameter $r = 0.87$. The x_n values move towards an attractor of period 4

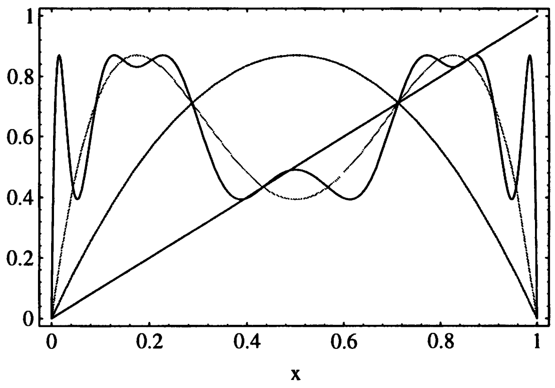

Fig. 3.2. The function $f(x)$ and its two- and fourfold iterations. The latter function has eight fixed points

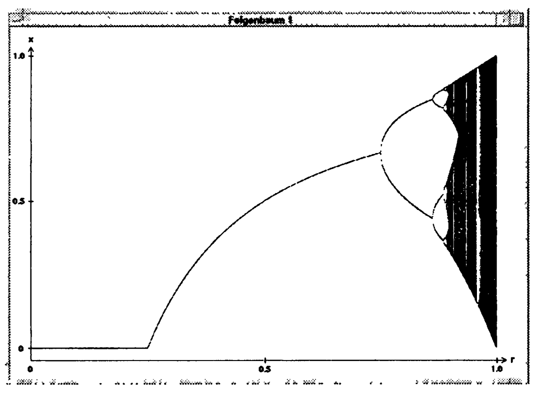

Fig. 3.3. Each vertical line contains 1000 x_n values from the iterated parabola, not showing the first 100 points. The parameter r varies between 0 and 1 along the horizontal

below a five-cycle; though there is just one band there, one still sees the five peaks of the adjacent cycle. It can be shown that only one band with density

$$\rho(x) = \frac{1}{\pi\sqrt{x(1-x)}}$$

exists for $r = 1$. In Fig. 3.4, in the chaotic region, one sees many windows with periodic orbits. The largest window, starting at the parameter $r = (\sqrt{8}+1)/4$ has a period 3. In this window, there is period doubling again: as r increases, one obtains the periods $3, 6, 12, 24, \ldots$ until a chaotic region appears. Indeed, it can be shown that all periods occur.

An additional impression of the logistic map $x_{n+1} = f(x_n)$ is obtained by geometrically constructing the sequence of points $(x_n, f(x_n))$, $(x_{n+1}, f(x_{n+1})), \ldots$ in the x–y plane. One gets each subsequent point by moving horizontally from the current point to the angle bisector $y = x$, and from there, vertically to the parabola $y = 4rx(1-x)$. Figures 3.6 and 3.7 show this construction, which is performed rather easily with *Mathematica*. The scaling laws (3.4) and (3.5) allow us to determine the universal Feigenbaum constant δ. To this end, we calculate the parameters R_l, at which there

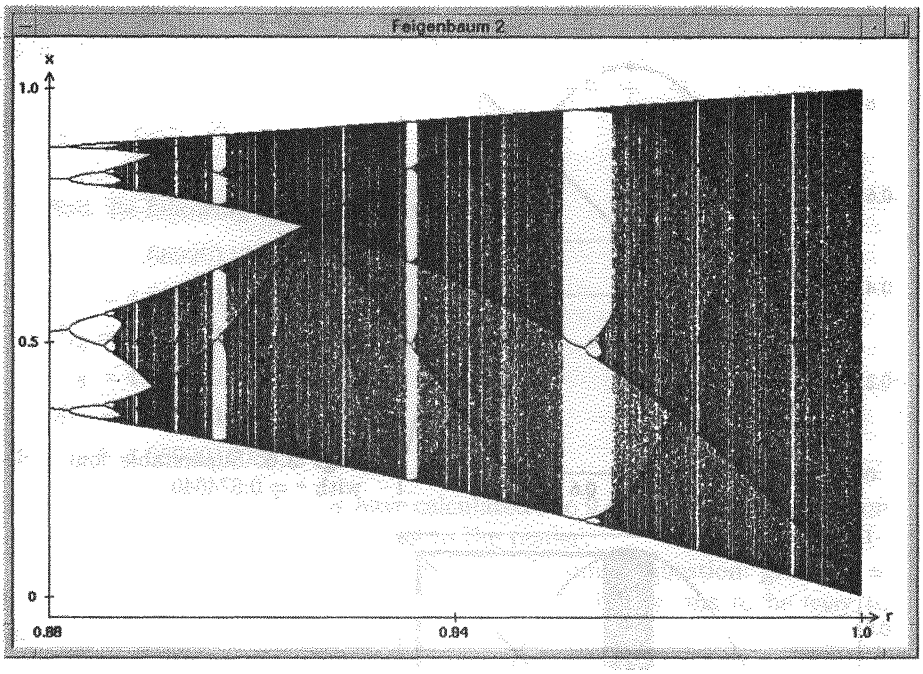

Fig. 3.4. A close-up from Fig. 3.3, for r between 0.88 and 1

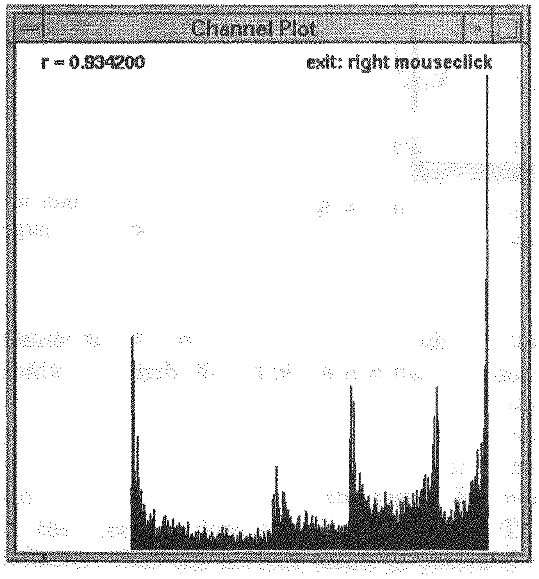

Fig. 3.5. Frequency distribution of the x_n values in the iteration of the parabola for a parameter value $r \simeq 0.934$, just below the window with period 5

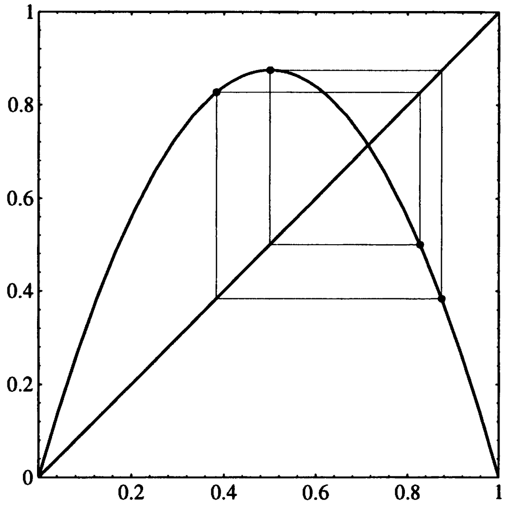

Fig. 3.6. Superstable four-cycle with $r = 0.874640$

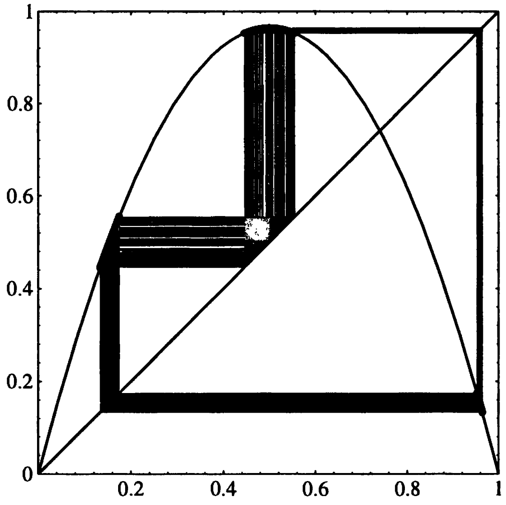

Fig. 3.7. Three chaotic bands for $r = 0.9642$, just above the large window with the three-cycle

are superstable orbits of period 2^l. Using the methods of symbolic dynamics and inverse iteration we find the R_l with a precision of 30 digits, of which only 12 are printed here:

$$
\begin{aligned}
R_{10} &= 0.892486338871\,, \\
R_{11} &= 0.892486401027\,, \\
R_{12} &= 0.892486414339\,, \\
R_{13} &= 0.892486417190\,, \\
R_{14} &= 0.892486417801\,, \\
R_{15} &= 0.892486417932\,.
\end{aligned}
$$

It is worth noting that the last value is obtained by solving an equation with a 32 768-fold nested root!

Using

$$\delta_l = \frac{R_{l-1} - R_{l-2}}{R_l - R_{l-1}} \,,$$

we find the following approximate values for δ:

$$\begin{aligned}
\delta_{10} &= 4.669201134601 \,, \\
\delta_{11} &= 4.669201509514 \,, \\
\delta_{12} &= 4.669201587522 \,, \\
\delta_{13} &= 4.669201604512 \,, \\
\delta_{14} &= 4.669201608116 \,, \\
\delta_{15} &= 4.669201608892 \,.
\end{aligned}$$

From this we estimate that we have calculated δ to 9 digits. The extrapolation of δ_l for $l \to \infty$ is left as an exercise to the reader.

Finally, we want to investigate a seemingly simple question: given $x_0 = 1/3$ and $r = 97/100$, what is the value of x_{100}? This seems to be easy to answer; with

```
h[x_]:= 97/25 x(1-x)
```

the command

```
Nest[h, N[1/3],100]
```

yields the value $x_{100} = 0.144807$. This result is wrong, however; the correct value is $x_{100} = 0.410091$.

The solution to this discrepancy lies in the chaotic behavior of the logistic map for the given parameter value. The Lyapunov exponent, defined in (3.6) to (3.8), is positive. Therefore, small inaccuracies in the calculation lead to large errors after just a few iterations. Owing to the command `N[1/3]`, *Mathematica* calculates everything to machine precision, which is 16 digits in our example. In each step of the iteration, the result is rounded; therefore the deviation from the true value, which slightly increases in each step, cannot be tracked. By using the format `N[1/3,prec]`, on the other hand, one can specify the precision via `prec`. Now *Mathematica* reduces the calculated precision after each step until, finally, the variable x_n is set to the value 0 when only one digit is left. The command `Precision[x]` asks for the precision of x. Figure 3.8 shows the result. For `prec=16`, each result is rounded, so the precision does not seem to change. The improved calculation shows, however, that the correct result is obtained only if each step is calculated with a precision of more than 60 digits. We do not know the reason for the linear relation between the precision of the calculation and that of the result.

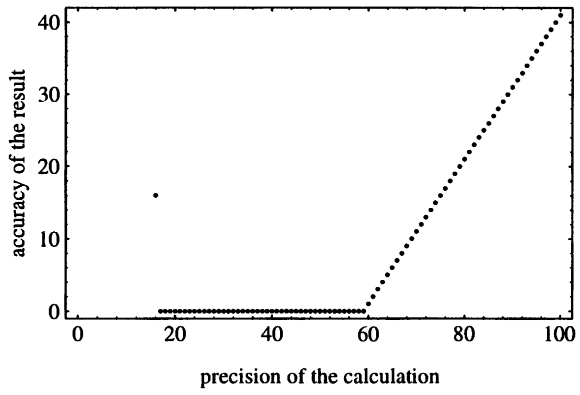

Fig. 3.8. The accuracy of x_{100} for $r = 97/100$ as a function of the precision with which the calculation is performed. The vertical axis gives the number of digits to which *Mathematica* calculates the result of the iteration

Exercises

1. Calculate the Lyapunov exponent for the logistic map (3.1) as a function of the parameter r.
2. In each r interval in the chaotic region, there are windows with superstable periodic orbits. Any such orbit starts with the symbolic dynamic *CRL*. Therefore, there are at most four possibilities for a five-cycle: *CRLRR*, *CRLRL*, *CRLLR* and *CRLLL*. One of these possibilities is not realized, however. Calculate the parameter values of the remaining three orbits of period five.

Literature

Crandall R.E. (1991) Mathematica for the Sciences. Addison-Wesley, Redwood City, CA

Bai-Lin H. (1984) Chaos. World Scientific, Singapore

Jodl H.-J., Korsch H.J. (1994) Chaos: A Program Collection for the PC. Springer, Berlin, Heidelberg, New York

Ott E. (1993) Chaos in Dynamical Systems. Cambridge University Press, Cambridge, New York

Peitgen H.O., Saupe D. (1988) The Science of Fractal Images. Springer, Berlin, Heidelberg, New York

Schroeder M. (1991) Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise. W.H. Freeman, New York

Schuster H.G. (1995) Deterministic Chaos: An Introduction. VCH, Weinheim, New York