Lab 5 - Nonlinear maps and chaos

Thursday 25 September - Due: Thursday 2 October

In this lab we will do some numerical experiments with perhaps the simplest non-linear system – the so-called *logistic map*. We will see that the long time behavior of this dynamical system falls into two possible classes – periodic behavior and chaotic behavior. Furthermore, the approach to chaos occurs via a *period doubling cascade*. In a chaotic regime the motion is confined to a *strange attractor* - a fractal object with non-integer dimension.

1 Logistic Map Dynamics

1. Download the code `logistic.py` from the LABS page and save it to your filespace. Run the code with `r=0.3`, a total number of iterations `MAX=20` and an initial `x=0.1` and note the sequence of values of `x` that are produced. Does the final value of `x` approach a fixed value ? Does this final value of `x` depend on its initial value ? Show results for differing initial `x` and fixed `r=0.3`.

2. Now fixing the initial value of `x=0.5` run the code for varying `r` in the range `0.25--0.7` in steps of `0.05`. You should change `MAX=10000`. Make a plot, using say Excel, of the exiting value of `x` versus `1/r`. What do you notice ? Can you find a formula for the final value of `x` as a function of `r` ? (Hint: If we approach a fixed value we can set `x` appearing on the lefthandside of the logistic map to its value on the righthand side. Use this in the formula for the logistic map to determine this *fixed point value*).

3. Now set `r=0.76` and run the code. Do you see a fixed point attractor solution at large times ? If not what is the period (repetition time) of the dynamics ? What distinct values of `x` are seen at large times ?

4. So for `r<0.7` one sees a fixed point but at `r=0.76` a different periodic behavior is seen. Can you locate the value of `r` to say 3 decimal places which marks the boundary of the two different types of behavior ?

5. Increase `r` a little until the period of the motion doubles again. Record the value of `r` at which this occurs to three decimal places.

6. Find the value of `r` at which transition to a 8 period motion occurs.

7. Use your numbers to estimate the Feigenbaum constant which is given by

$$\delta = \frac{r_4 - r_2}{r_8 - r_4}$$

   where r_2 corresponds to the value of `r` at which a period 2 motion first occurs, r_4 a period four motion etc.

2 Bifurcation diagram

1. Clearly it would be nice to try to automate this process of finding values of `r` at which period doubling occurs. Download the modified code `bif.py` from the LABS page. This code loops over a `list` of `r` values generated by the `arange()` function and for each value of `r` iterates the logistic map `MAX` times before recording the final `64` exiting values of `x`. What is the maximum period which can be resolved with this code ? The code plots a picture showing the possible values of `x` at each `r` value. This is called the *bifurcation diagram* for the map.

2. It should be clear that for `r` values bigger than 0.9 or so the system is chaotic. Can you see any evidence for regular behavior at large `r`? You should be able to locate a stable period 3 motion at large `r`. Make a screen capture of your bifurcation diagram.

3 Fractal dimensions

1. Let us now investigate the effective dimension of the set of points produced by the dynamics in the chaotic regime. Download the code `dimension.py` from the LABS page. This code divides the line interval $0 \rightarrow 1$ into $N = 2^{\mathtt{P}}$ segments each of length $s = 1/2^{\mathtt{P}}$ and runs the logistic map dynamics collecting how many points n_i fall into the segment labeled by i. It then performs the sum

$$T(s) = \sum_{i=1}^{N} n_i^Q$$

   Finally it plots out `log(T(s)) vs log(s)` – this should be a straight line whose gradient is the effective dimension. For periodic motion a finite set of points is produced whose effective dimension is zero. If the points covered the interval uniformly and randomly one would see a gradient of unity (the dimension of a regular line). In the case of chaos one should fine a line with slope less than one – the fractal dimension of the strange attractor governing the chaotic motion. Initially set `r=0.5` and `Q=0` and run the code. Estimate the slope.

2. Now set `r=0.896` and rerun the code (still with `Q=0`). Estimate the slope (and cut/paste a screen shot of what you see into your lab report)

3. Modify the code to output the pairs of numbers

$$(-\mathtt{log}(\mathtt{box_length}), (\mathtt{1}/(\mathtt{1.0} - \mathtt{Q})) * \mathtt{log}(\mathtt{count}))$$

   to successive lines of the interactive IDLE window. Cut and past these into Excel and use the latter to fit the data to a straight line. In this way you should be able to determine the effective fractal dimensions more accurately. Include the modified code in your writeup.

4. Keeping `r=0.896` record the various effective dimensions corresponding to `Q=0,2,4`

4 Sierpinski dynamics

1. Download the code `sierpinski.py` from the LABS page and run it. We will have more to say about this next week. Make a screen capture for this week's lab of what you see.