

# Lec6

- Chaos - logistic map
- Period doubling, strange attractors, fractals
- Sierpinski triangle, chaotic dynamics
- Fractal dimension

# Logistic Map – lab 5

- Simplest example of chaotic dynamical system
- Exhibits period doubling approach to chaos
- In chaotic regime motion confined to *strange attractor*
- Fractal object - dimension *non-integer*

# Model

Model for population growth after  $n$  steps of reproduction.

Let  $P_n$  represent population in generation  $n$

$$P_{n+1} = P_n(a - bP_n)$$

- $a$  represents unlimited reproduction rate
- $b$  represents competition limited growth

Rescale:

$$x_{n+1} = 4rx_n(1 - x_n)$$

Single parameter  $r$  controls dynamics. To keep  $x_n$  positive impose  $0 < r < 1$  and  $0 < x_0 < 1$

# Dynamics

- Final state at large times independent of initial state
- For small  $r$   $x_\infty = 0$
- For  $r < 0.75$   $x_\infty = 1 - \frac{1}{4r}$
- For  $r$  a little above 0.75 see period 2 motion.
- Continues. Above  $r = 0.86$  see period 4 motion etc
- Period doubling continues until *at some finite*  $r = 0.892..$  motion becomes chaotic. Change in  $r$  required to double period *universal* Feigenbaum constant

# Strange attractors

- In chaotic regime values of  $x$  *never* repeat. Motion looks random yet *cannot* be.
- Some regions in  $0 < x < 1$  *never* visited!
- Set of points is a *fractal*. Such an object looks same under magnification.
- Not a standard geometrical object - has a non-integer effective dimension.
- Note: independent of  $x_0$  dynamics leads to motion on this fractal *strange* attractor

# Fractal dimensions

For a regular object can define the dimension of the object of linear size  $R$  from the relation

$$M(R) \sim R^D$$

or

$$D = \frac{\ln M(R)}{\ln R}$$

- Can use this to define/calculate dimension for a fractal
- *Cover* fractal by a grid/lattice of cells
- Figure out how many cells are required to cover the fractal as a function of the size of the cells
- Use this in relation like above to compute  $d_F$

# Logistic Map attractor

- Points on attractor live in  $0 < x < 1$
- Divide this segment into  $2^P$  equal pieces.
- Count how many points lie in each cell
- Define (one) dimension by plotting number of cells needed to cover fractal against length of cell.
- Gradient of straight line =  $d_F$

# Comments

- Notice  $d_F < 1$ . Does not fill embedding space! Holes of all sizes seen. Fills vanishing fraction of all points in  $0 < x < 1$ !
- Infinite number of points on fractal – but represent a vanishing fraction of all points in  $0 < x < 1$ . Like eg. number of rational numbers  $p/q$ . Infinite in number but a vanishing fraction of all *real* numbers.
- Other definitions of dimension possible. Multifractals.

# Many dimensions

- Can define many dimensions this way. Suppose iterate dynamics  $N$  times. Calculate number of points  $n_i$  in cell  $i$  with scale factor  $s$

- Compute

$$d^Q = \frac{1}{Q-1} \frac{\log\left(\sum_i^{N(s)} n_i^Q / N\right)}{\log s}$$

- $Q = 0$  box counting dimension just discussed
- $Q = 2$  mass dimension introduced earlier
- $Q = 1$  exists and is called *information* dimension.

# Other fractals - Sierpinski triangle

- Example of regular fractal. Looks *exactly* the same on all scales.
- Can be defined *recursively*. Exploits self-similar nature of fractal.
- But can also be seen as the strange attractor of a special nonlinear dynamics.
- Exhibits a fractal dimension  $d_F = \log(3)/\log(2)$

# Sierpinski dynamics

Points  $(x, y)$  on triangle originate from dynamics

$$x = ax + by + e$$

$$y = cx + dy + f$$

where set  $(a, b, c, d, e, f)$  comes in three flavors. Which set is used for a given update is chosen at random

This is how what looks like a linear update becomes effectively a *nonlinear dynamics*

# Calculating the dimension of regular fractals

eg **Koch curve**:

Start from line; add triangular bump

then add add bump to all sublines etc

At each stage number of line segments needed  
goes up by 4

scale distance goes down by factor of 3

Thus  $d_F = \frac{\log 4}{\log 3}$

**Sierpinski** similar:

Each iteration needs 3 more triangles to cover  
object

scale length down by factor of 2.