Complexity in social dynamics : from the micro to the macro
Lecture 2

Franco Bagnoli

Namur 7-18/4/2008

Motivations

Psychology, sociology, linguistics, literature, politics, arts. . . something in common?

- They are different "views" of the same topic: human brain, how it works, how it communicates with other brains, how all these brains self-organizes.
- From single neurons to society there are several "description" levels.
- As usual in hard sciences (from elementary particle physics to atomic physics, matter physics, chemistry, engineering. . .) , we recognize a sort of "separation" among levels.
- Each level has its own time-scales. Averaging over these scales one obtains the effective "rules", to be used at the subsequent level..

Is the complexity in nature or in our brain?

- Some topics (those studied at school) are simple, others (in real life) are complex.
- Understanding is similar to playing games: extract the rules (modeling) and become a master (training).
- Training requires brain structures that are presumably always the same. They are good for dealing with a few objects at a time, and few nonlinear "exceptions".
- At school, beyond training in playing games (math), one learns techniques for reducing a problem to a small number of variables. (scale separation, collective modes, etc.)
- Problems that cannot be reduced are considered "complex". They may nonetheless be simulated on a computer. Algorithmic complexity is different from human complexity.

Why simple modeling

- In order to learn reduction techniques (also when they do not apply), we do not need systems formed by complex units, only systems formed by *many* units.
- It is much easier (and clearer) to use simple building blocks.
- It allows to use also non-sophisticated numerical techniques and environments.
- Clearly, in real life one has to resort to number crunching, grid computing, code optimization, careful model design...

Summary

1. Dynamics and noise.
2. Discrete systems: cellular automata. Trajectories, attractors and basins.
3. Chaos and the stochastic approach. Probability distributions.
4. Stochastic automata. Transition probabilities. Observables and control parameters.
5. Markov processes and phase transitions.
6. Transients.

Three very very simple models

To start, let us consider a game:

- We have a ring of chairs, oriented towards the exterior of the ring. People can only see the nearest neighbors.
- Each individual has a flag, that can be up or down.
- At a gong signal, all take into consideration the number of up flags among themselves and the neighbors, and rises or lowers (or keeps) the flag according with the rule.
- The first version of the game is: if the number of up flags is odd, rise flag, otherwise get it down.
- The second version says: follow the majority.
- The third version is: if any of your two neighbors' flag is up, then pick a card from a deck, and if the card shows a flag then rise the flag (each one is given a deck).

Dynamical systems

- We can assume to be able to describe the behavior of a microscopic system by a deterministic rule (say, Newtonian Dynamics) even though at a quantum level nature is probabilistic.
- This assumption is sometimes true also at a higher level, for instance fluid dynamics is described by deterministic equations.
- What we call "noise", and that leads to stochastic dynamics, is originated by the partial knowledge of the state of the system, and by the "mixing" and amplification of this uncertainty: chaos.
- The systems that we study are composed by many interacting elements. (*extended systems*). Let us introduce some terminology.

Discrete systems and automata

- We shall start considering discrete systems, since they are *exactly* computed on a computer.
- By “discrete” I mean that time, space and state variables take only integer values. These systems are known as *cellular automata*.
- An automata is a discrete dynamical system characterized by an internal state (integer-values), an output or visible state that is a function of the internal one, and a rule for changing the internal state as a function of the internal state itself, and (optionally) some other input.
- For instance, a compiler is an automata that translates a stream of symbols (the source code) into another sequence of symbols (the machine code).

Cellular automata

- A cellular automaton (CA) is a network of automata, generally extremely simplified.
- It is defined by the network of connections (links) among elements (nodes or "cells" for a regular lattice), the number of possible states (internal and external states generally coincide), and the update rule.
- Let us number the nodes with the index i, denote by $x_i(t)$ the state of node i at time t, and indicate the set of the states of nodes connected to i with $V_i(t)$ (vicinity or neighborhood), including the node itself.
- The evolution rule for a single node is

$$x_i(t+1) = f(V_i(t)).$$

- In the following we shall neglect to indicate the time t, and use a prime to indicate the state at time $t+1$, so we have

$$x_i' = f(V_i)).$$

Evolution rule

- The evolution rule f is a discrete function of discrete variables.
- The possible input configurations are finite. It is possible to code them as integers (for instance, using base-two numbers for Boolean automata).
- Therefore, the function f can be specified using a string of numbers, or again, just a number.

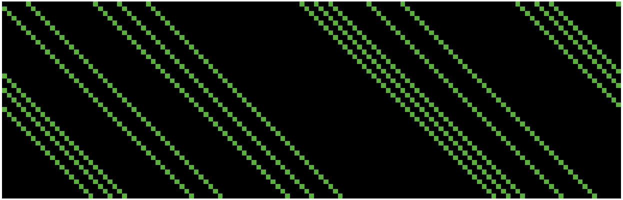
inputs			input code	output	output weight
0	0	0	0	0	1
0	0	1	1	1	2
0	1	0	2	1	4
0	1	1	3	0	8
1	0	0	4	1	16
1	0	1	5	0	32
1	1	0	6	0	64
1	1	1	7	1	128

total: rule 150

Examples 1

Let us start with some simple examples for Boolean automata ($x_i \in \{0, 1\}$), regular, one-dimensional lattice and neighborhood $V_i = (x_{i-1}, x_i, x_{i+1})$. These are called *elementary cellular automata*.

- The rule $x'_i = x_i$ is clearly the identical rule (170).
- The rule $x'_i = x_{i-1}$ is the right-shift rule (241).

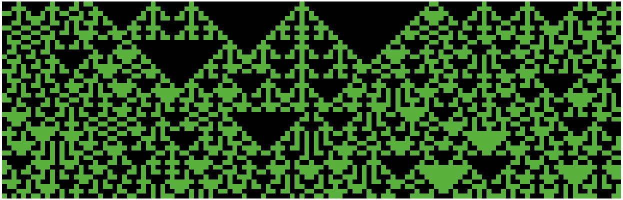


Notice the use of periodic boundary conditions. Time goes downward.

Examples 2

- Elementary CA may lead to “simple” dynamics (majority rule 232)

- Or “chaotic” ones (rule 150)



Trajectories

- The state of a the whole system ($\mathbf{x}(t)$) can be characterized with a single number (for instance by reading the configuration in base-two for a Boolean automata

$$\mathbf{x}(t) = (x_i(t))_{i=0,\dots,N-1} = \sum_{i=0}^{N-1} x_i(t) 2^i$$

- The sequence of the states generated by the dynamics is called a *trajectory*
- Since the dynamics is deterministic, after that a state appears twice in a trajectory, the system has entered into a *cycle* whose length is called *period*. Period-one cycles are called *fixed points*.
- The interval before entering a cycle is called *transient*.

State space

- The set of all possible configurations is called the state space (2^N for Boolean CA).
- The cycles (and fixed points) are called *attractors*.
- The set of configurations that are mapped on an attractor by dynamics is the *basin* of the attractor.
- The phase space is partitioned into one or more basins.

Back to examples

- The phase space of identity rule 170 is formed by 2^N fixed points. No transients.
- The phase space of the shift rule 241 is formed by 2^N many cycles of period N. No transients.
- The phase space of the majority rule 232 is formed by many fixed points (all configurations with at least two consecutive zeros or ones), with short transients and small basins.
- The phase space of rule 150 is formed by one or more very long cycles, with period that scales exponentially with N (and therefore large basin). There is also at least one fixed point: state 0, with a very small basin (only itself, I think).

Chaos and classification

- One can classify configurations according with the basin they belong to.
- This is particularly useful is attractors are fixed points. In this case the dynamics "classifies" the initial states (example: rule 232).
- The opposite situation is when there is a single cycle spanning all states, or, almost equivalently, when the period of the largest cycle grows exponentially with the system size. In this case it generally happens that a small "error" propagates to all the system (chaos).

Observables

- An observable A is a function (scalar or vectorial) of a configuration $A(t) = A(\mathbf{x}(t))$ (a projector). It corresponds to measurements.
- If the observable A is given by the average of local quantities, it is called a *extensive* quantities.
- The sequence of values of $A(t)$ is a *time-series data*.
- In general, by projecting one looses short-time determinism.
- However, it is possible to reconstruct the topological properties of the attractor (fractal dimension) from the time-series data by plotting delayed data (embedding). This requires very long time series with the same conditions.
- For short time-series, one may introduce a stochastic dynamics.

Stochastic automata

- The evolution rule may be made stochastic, by defining the probability $\tau(x|V)$ of getting x given a certain neighborhood configuration V.
- Deterministic CA can be seen as particular cases of stochastic CA for which the transition probabilities τ are either zero or one.
- For the implementation, one may think to having the time space lattice defined at beginning, and that there are random numbers $r_i(t)$ (uniformly distributed in $[0, 1[$) attached to sites (quenched field).
- The evolution rules depends on the random numbers, for instance, *directed percolation*:

$$x_i(t+1) = (x_{i-1}(t) \vee x_{i+1}(t)) \, [r_i(t+1) < p]$$

- Where p is a probability, the symbol $\vee$ is the Boolean or, and $[\cdot]$ is the truth function.

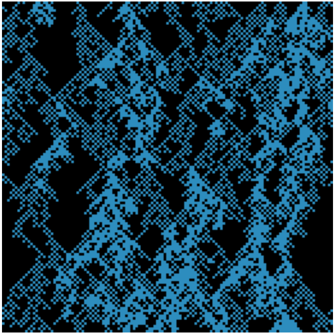
Stochastic dynamics

- Once that the random numbers $r_i(t)$ are set, the dynamics is deterministic, and we can use the previous definitions.
- In general, one is interested in the *average* behavior (over the *realizations of nose* $r_i(t)$.
- The probability p is called an *control parameter*.
- By changing p one may observe sudden changes in the asymptotic behavior of the system (change of stability in the attractors).
- For directed percolation, there is always an attractor $x = 0$, whose basin extends to all configurations (for finite systems).

Transients

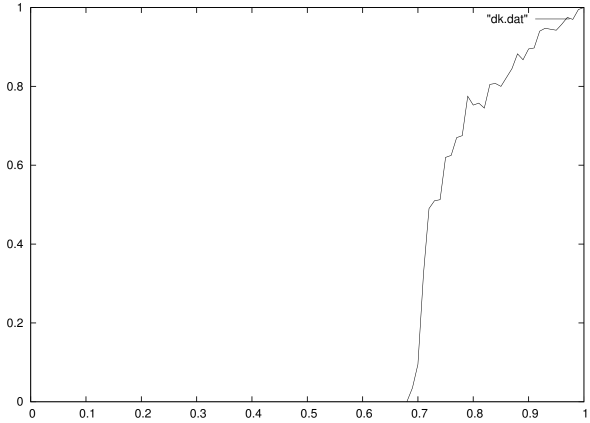
- For small p (here 0.65) transients are small

- For $p = 0.7$ transients may scale with system size (exponentially)



Phase transitions

- For large systems ($N \to \infty$) the asymptotic configuration changes from 0 to an "active" state, with nonzero density of nodes with value 1



Probabilistic approach

- The stochastic dynamics, after averaging over the disorder (or over replicas), leads to probabilistic considerations.
- Let us call P(x, t) the probability of observing the configuration x at time t. It is defined by averaging over K replicas

$$P(\mathbf{x}, t) = \frac{1}{K}\sum_{k=1}^{K}[\mathbf{x}^{(k)}(t) = \mathbf{x}]$$

  for $m \to \infty$.
- The probabilistic evolution rule defines a Markov chain M

$$P' = MP,$$

  where $\mathbf{P} = P(\mathbf{x}, t)$ and $\mathbf{P}' = P(\mathbf{x}, t + 1)$.
- A phase transition corresponds to the degeneracy of the largest eigenvalue ($\lambda = 1$) of M.

Thermodynamic limit

- The Perron-Frobenius theorem says that for an irreducible Markov matrix the largest eigenvalue is not degenerate.
- The irreducibility implies that there is a possible chain that leads from any state to any state
- Even in the presence of absorbing states (configuration 0 for the directed percolation), the matrix is irreducible. The degeneracy comes from the limit $N \to \infty$ (taken before $T \to \infty$): the so-called thermodynamic limit.