RANDOM SEQUENCE GENERATION BY CELLULAR AUTOMATA

Scientific Computing II
Fundación Universitaria Konrad Lorenz
February 24, 2024

You are expected to communicate accurately the solution of the exercises in this assignment. Full score will only be given to correct and completely justified answers. Miraculous, obtuse and unnecessarily complex solutions will receive partial or null score. You can ask any question of this assignment during class or through email: julian.jimenezc@konradlorenz.edu.co.

The deadline to submit this assignment is **March 02, 2024, 8.15am.**

This assignment aims to guide you to understand how pseudo-random numbers are generated.

Note: This homework is based on Wolfram's article on how to generate pseudo-random numbers using cellular automata.

Random numbers are extremely important for pure and applied math. Nevertheless, its generation is hard. In order to generate consistent random numbers, the algorithm doing that has to pass many randomness tests in order to be applicable. Nevertheless, computers cannot generate random numbers from the void: all algorithms require a *seed* that initializes the generator, so that if the seed we use is the same, we will retrieve the same *random* numbers. That's why generated random numbers are usually called **pseudo-random numbers**.

A way to generate random numbers is using rule 30, which is a 1D cellular automata model (see Figure 1). In this assignment, you are expected to program a Python script able to generate pseudo-random numbers and then test that they are "random" using a histogram.

1. (0/20) Let $\boldsymbol{a}$ be a 1-dimensional array containing 1 or 0, such that cell i is alive (dead) at time t if $a_i(t) = 1$ (or 0, respectively). Show that the set of rules of Rule 30 is equivalent to the following discrete dynamical system.

$$\begin{aligned} a_i(t+1) &= a_{i-1}(t)\,\mathrm{XOR}\,(a_i(t)\,\mathrm{OR}\,a_{i+1}(t)) \\ &= (a_{i-1}(t) + a_i(t) + a_{i+1}(t) + a_i(t)a_{i+1}(t)) \bmod 2. \end{aligned} \tag{1}$$

2. (0/20) Write a Python script that evolves Rule 30 for a single alive cell at $t = 0$ till $t = 1e6$ (one million steps). You are expected to vectorize your code and it should run smoothly and fast (optimize your code!).

3. (0/10) From the previous item, you should have the values of the central column (where was the alive cell at $t = 0$). It is said that this column is chaotic. Use it to generate random numbers from 0 to 7, and then graph a histogram to study if the appearance of these numbers is uniform.

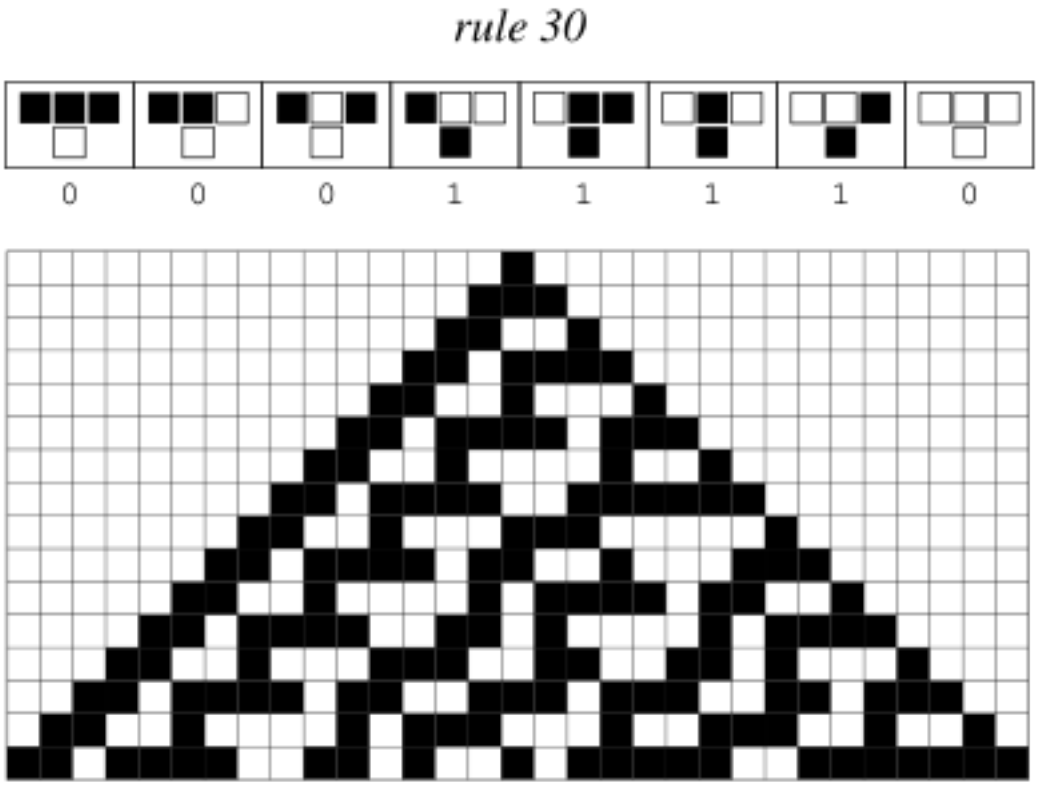

FIGURE 1. The upper row of this figure represents how this rule works. For example, the first rule implies that three alive cells will lead to a central dead cell (overpopulation). On the other hand, the grid below the rules represents how the system evolves, starting from the first row, and ending in the last row, for a single initial alive cell.