Gaussian Continuous Cellular Automaton

Samuel Landis - Computer Science, University of New Mexico
Enrique Almeida Ruiz - Mathematics, University of New Mexico

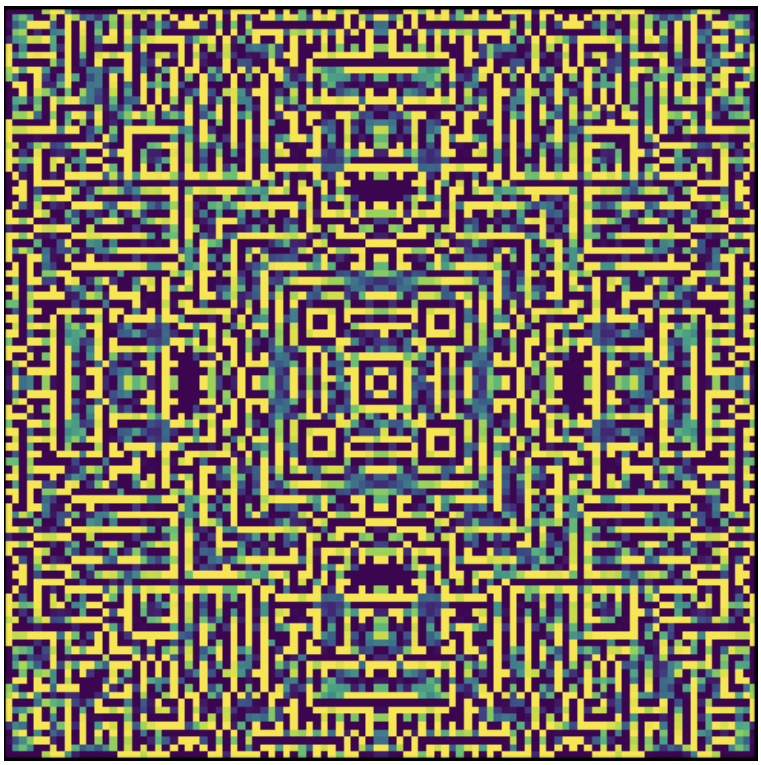

Figure 1. Simulation with $x = 1/10e$

Abstract

This project investigates how mathematical stability affects numerical computations by exploring a system of cellular automata using a Gaussian-inspired update function. Specifically, we examine how variations in the input parameters influence the variability of results across repeated calculations. The system evolves through stochastic interactions, where numeric instability causes divergence over time. As with other cellular automata, small differences in starting states can lead to significantly different outcomes as the system progresses. This is a phenomenon that we aim to quantify. By measuring variability, defined as the difference between the maximum and minimum living cell counts at a predetermined time across multiple runs, we seek to reveal how seemingly minor computational inaccuracies can propagate and amplify in dynamic systems. This study highlights the subtle yet critical role of numerical precision and stability in simulations influenced by emergent behavior.

Introduction

Cellular automata are simple yet powerful systems where complex behavior emerges from simple rules. Originally, our plan was to extend Conway's Game of Life by introducing fractional states, where each cell's state ranges continuously between 0 and 1, representing how "alive" it is rather than being strictly alive or dead. This was intended to be a better model of real-world phenomena. This would also allow us to examine how limited numerical precision affects simulations. While experimenting with update functions, we observed unexpected behaviors tied to stability. Small numerical differences accumulated over time, causing runs that began identically to diverge. Symmetric simulations would suddenly fall into chaos. By repeatedly running simulations with identical starting conditions and measuring the variability in living cell counts, we aim to understand and quantify how minor numerical inaccuracies propagate.

System Design

Our continuous cellular automaton is implemented on a square two-dimensional grid of cells that are represented by floating-point values ranging between 0 and 1. This continuous state was chosen over a more discrete binary state to better simulate natural phenomena. The automaton evolves in discrete time steps, and at each step, every cell updates its state based on the sum of its neighboring cells' states using a Gaussian-inspired rule. For every cell, we calculate the sum of the eight neighbors that it touches. This sum represents the local environment's influence. The change in the cell's state is determined using the following formulas:

$$\text{cell value}_n = \text{cell value}_{n-1} + \Delta$$

Equation 1. Next Cell Value Function

$$\Delta = 0.1 \cdot \left(e^{-(\text{sum of surrounding cells} - 3)^2} \right) - x$$

Equation 2. Cell Change Function

The cell's new value is calculated by adding the output of the change function (Δ) to the cell's current value. Boundary conditions are enforced to keep all states within the range [0, 1]. Any value calculated below 0 is clamped to 0, and any value above 1 is clamped to 1.

x is a tunable parameter controlling the system's behavior. The Gaussian-like function peaks when the sum of the surrounding cells is near 3. Changing *x* changes the growth or decay tendency of the cells. Additionally as we shift the Gaussian function by decreasing *x*, we restrict the range where the cells will continue living. This allows us to fine tune the simulation as shown below.

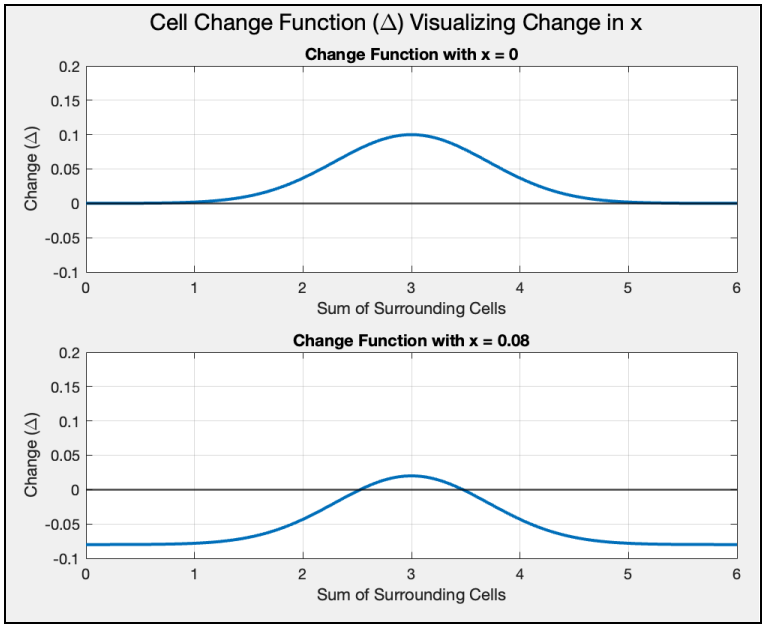

Figure 2. Change Function (Δ) with different *x* values

Experimental Setup

To examine the effects of *x* on stability, we performed repeated simulations with different values of x and then compared this to the variability of the end state of the simulation. We used the sum of the values of all cells at a predetermined time step as the end state of the simulation. We then calculated the average and standard deviation of those sums. For each value of *x* tested, the automaton was initialized using a randomly generated starting state with a size of 20 cells by 20 cells and run 10 times for a 1000 steps. We tested many different values of *x* between *x* = -0.005 and *x* = 0.08. The results for the average (mean) and standard deviation are shown below.

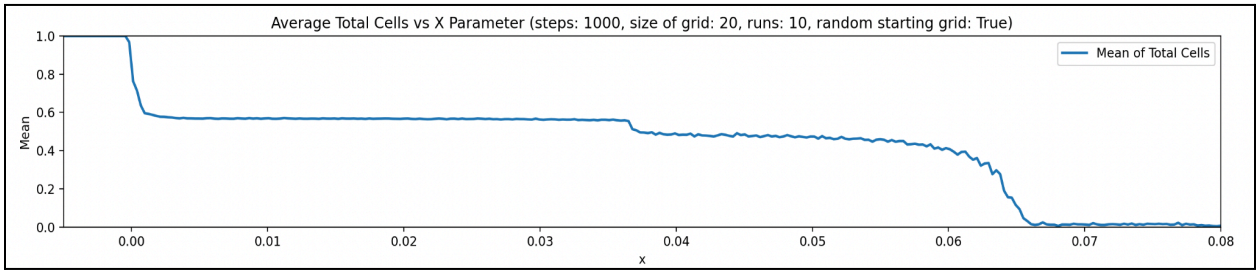

Figure 3. Mean of total cells for different values of *x* using random starting state

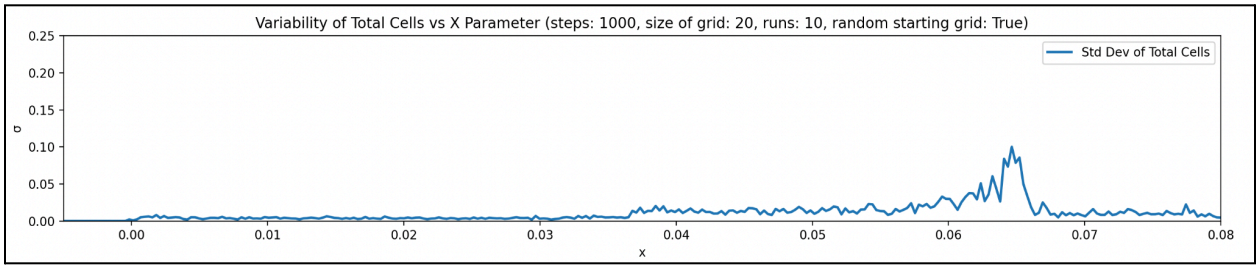

Figure 4. Standard deviation of total cells for different values of *x* using random starting state

Results

Between $x = -0.005$ and $x = 0.0001$ the cells all converged to a value of 1. Between $x = 0.0001$ and $x = 0.0366$ the cells all converged to a stable state similar to the first image in the image gallery at the end. The value 0.0366 is interesting. Using manual testing we narrowed this value down to approximately 0.0367879441171442, which is the same as 1/10e or 0.1 * e. Between $x = 0.0366$ and $x \approx 0.05$ the simulation was quite lively but not stable by any means. The patterns that emerged weren't static or coherent. They pulsed and folded in on themselves. They looked almost psychedelic similar to the second image in the gallery. Between $x \approx 0.05$ and $x \approx 0.065$ the simulation decayed, unraveling into chaos. This is illustrated by the third image. After $x \approx 0.065$, the simulation basically died out or ended up as 2 by 2 squares of cells. This was the only stable pattern that was discovered through this project. These can be seen in the fourth image. When x was varied mid-simulation, the most interesting patterns emerged as can be seen by the rest of the images in the gallery.

Conclusion

This study demonstrates how continuous cellular automata, governed by a Gaussian-inspired update function, can act as sensitive indicators of numerical stability in simulations. By introducing a tunable parameter x, we explored how small numerical differences, which may arise due to finite precision or chaotic dynamics, can propagate and amplify over time. Our results show that for certain values of x, the system stabilizes or converges, while for others, it devolves into chaotic or psychedelic patterns before collapsing into inactivity. Particularly noteworthy is the apparent critical point around $x = 1/10e$, where the behavior of the system begins to shift dramatically. This underscores the importance of numerical precision and parameter sensitivity in computational modeling.

Code: https://github.com/samuel-landis/MATH_375_Project

Image Gallery