Bak-Tang-Wiesenfeld Sandpile Model
Graduate Project

Blair Gemmer

May 9, 2012

1 Introduction

The main goal of this project was to explore the naturally ubiquitous "$1/f$" noise and to simulate it using the famous Bak-Tang-Wiesenfeld (BTW) sandpile model [1]. This "$1/f$" noise is something that BTW hypothesized was the reason for the phenomenon of cosmic strings, mountain landscapes, coastal lines, and other self-similar fractal structures. Their sandpile model shows how dynamic systems naturally evolve based on reaching some critical unstable state and the "domino" effect of the noise propagating through the unstable regions. They go on to say that this explains how turbulence is essentally long-wavelength perturbations causing cascades of energy dissipation on all length scales. In other words, everything that flows in some way or another will display this "$f^{-\beta}$" noise.

The interesting part of the BTW model is that it doesn't require any parameters. Compared to many physical models which are based on specific parameters such as temperature or velocity, this model is parameter-less which allows the system to evolve until the entire lattice is stable. At any point, if a single element in the lattice is unstable, the entire lattice becomes unstable from that point outward. This instability has been coined a "critical slope value" or an "angle of repose". Anything that reaches or somehow excedes this value will cascade downward, affecting all neighbors.

Therefore, the model can be represented as some cellular automaton, where each cell of the automaton matrix represents some sort of scalar slope $s_{i,j} = 0, 1, 2, 3, 4, 5, 6, 7$ at each column position i, j. The column i, j is unstable if $s_{i,j} > 3$. The sandpile is unstable if any column is unstable. The sandpile is updated with the following local rule:

if $s_{i,j} > 3$, then remove 4 units of slope from column i, j and add one unit of slope to each of the 4 neighboring columns:

$$\begin{aligned} slope_{i,j} &\to slope_{i,j} - 4 \\ slope_{i+1,j} &\to slope_{i+1,j} + 1 \\ slope_{i-1,j} &\to slope_{i-1,j} + 1 \\ slope_{i,j+1} &\to slope_{i,j+1} + 1 \\ slope_{i,j-1} &\to slope_{i,j-1} + 1 \end{aligned}$$

This rule is applied synchronously to all columns in the pile, and repeated until a steady state is reached.

Fig. 1 shows the steady state of a pile with 200x200 columns starting from an initial unstable state with $s_{i,j} = 7$. From the figure, it is clear that a steady state of this model can be extremely complex and intricate. The state is critical, which means that if a single grain ofsand is added to it, an avalanche of toppling columns is triggered, and the system settles down to a different critical state.

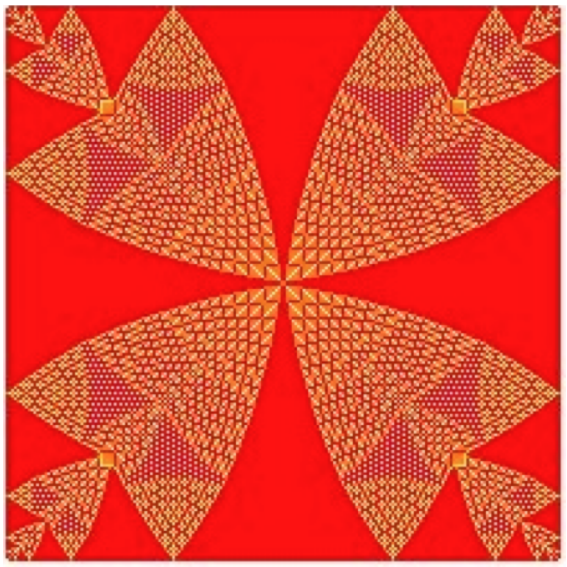

Figure 1: Critically Stable State ($n = 200$)

These events are so complicated that it is impossible to predict what will happen without actually running the simulation.

Let n_t be the number of topplings in an avalanche. By generating a large number of avalanches, one can measure the distribution of numbers $N(n_t)$ of avalanches as a function of avalanche size n_t. For the sandpile model, the distribution has a power law behavior:

$$N(n_t) \sim \frac{1}{n_t^b} \tag{1}$$

where b is the exponent of the power law. Power-law behavior implies that the avalanches have no natural scale or size: events of all sizes can occur in the distribution. If the exponent b is very small, then events of all sizes are equally probable. If $b > 0$, then larger events are less likely than smaller events. The special characteristic of a power law is that the ratio of events which differ in magnitude by a fixed multiplicative factor is independent of the size of the events. If the factor is 10, for example, then

$$\frac{N(10n_t)}{N(n_t)} = \frac{1}{10^b}$$

For the sandpile models, the exponent is found to be of order one: $b \simeq 1$. BTW called this behavior "Self-Organized Criticality". In other words, the sandpile evolves or organizes itself without any external influence into a complex critical state where events of all sizes can occur.[3]

A really easy way to envision this is an hourglass, with a grain of sand being dropped at regular intervals onto the top of the pile. Eventually, the slope of that spot on the pile will reach a critical point and the pile of sand will cascade downward in an avalanche. The same thing can be seen naturally occuring in mountain ranges and deserts as wind and rain shift the grains of sand and rocks around to form critical states of existence.

2 Method

The project utilized a python script to do calculation, animation, and overall plotting of the sandpile model. All graphs and animations were done using the loglog() plot in matplotlib. The algorithm differed slightly from the BTW model in that it didn't check the stability of the entire sandpile. Rather, it utilized a "flow" model, where the boundaries were closed and a single grain was added at random or specified points during each iteration.

There were different modes for initialization (called Init Mode in the diagrams) and the flow (called Dribble Mode). The sandpile could be intialized as random float values, random integer values, static value of zero, static value of 3 (near critical value), or a static value of 7 (over critical value). The pile could then dribble either randomly, statically in the middle, or statically on the 4 corners.

3 Verification of Program

Based on the paper by BTW [1], fig. 2 shows the power law distribution of the size of an event versus its frequency of occurring.

Likewise, when looking at the frequency of earthquakes (fig. 3), a similar power law behavior is exhibited.

Similarly, this project found that a power law behavior was exhibited over a long enough number of iterations (figs. 4, 5, and 6)

The animation for this project started by watching the behavior of the Sandpile applet [5]. Armed with the knowledge of how the sandpile was supposed to act, creating the simulation was relatively easy. A good portion of time was spent tweaking parameters and saving figures at each iteration to see the change at various times during the experiment. A good portion of time also was spent trying to get the animation working in openGL. The main problems occurred with trying to call glDrawPixels and using

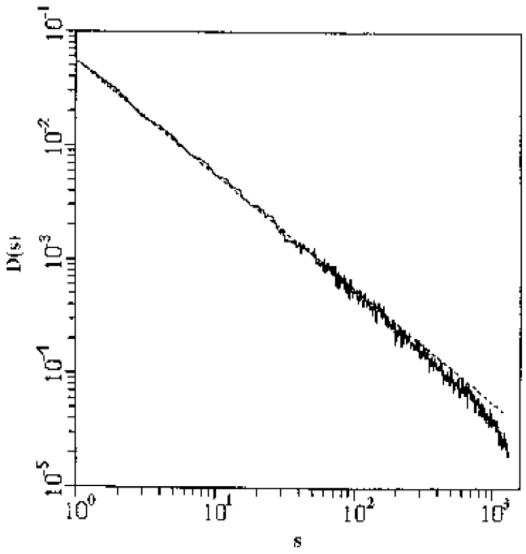

Figure 2: Domain Frequency Relation

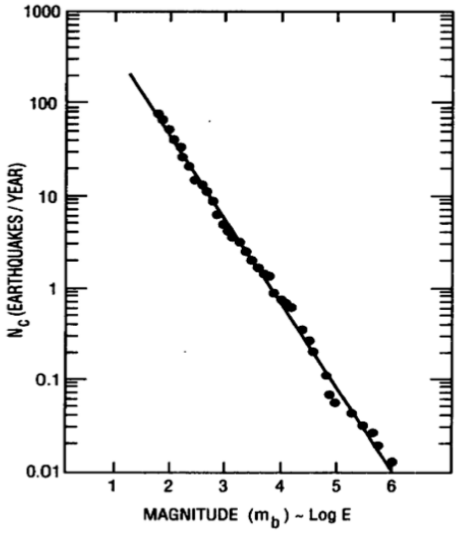

Figure 3: Earthquakes per annum vs. Magnitude of Earthquake

the wrong type of array. glDrawPixels expects an array of the type specified when calling the function. Numpy arrays are by default float64, but glDrawPixels defaults to unsigned byte, so this causes a conflict [6]. There ended up being enough problems and too much time was spent on debugging, so openGL was dropped from the project.

It is interesting to see the symmetry in the resulting figures from the animation (figs. 7-14). These are just some of the incredibly complex states of the evolution of the sandpile.

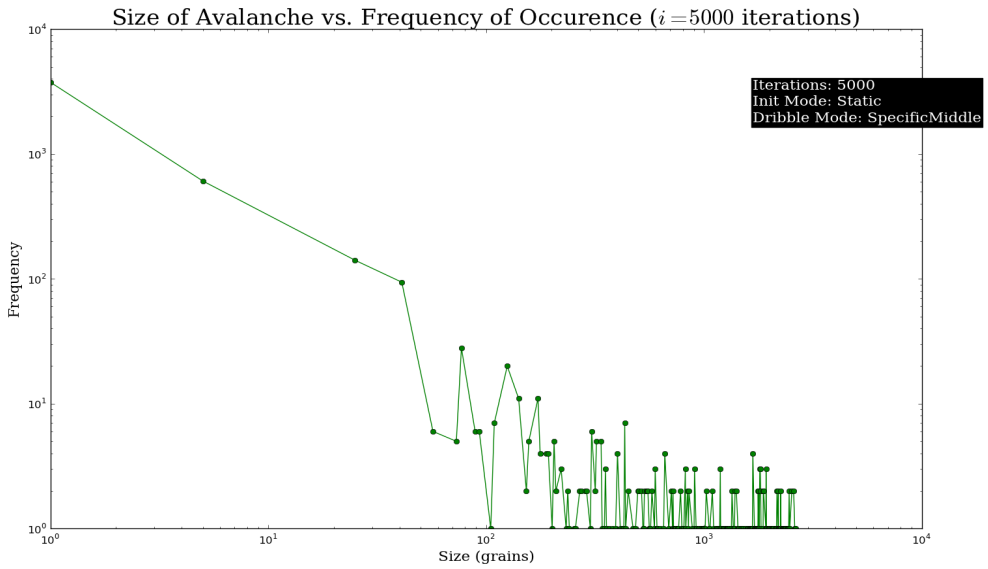

Figure 4: Size of Avalanche vs. Frequency of Size (Initialized at $s_{i,j} = 0$)

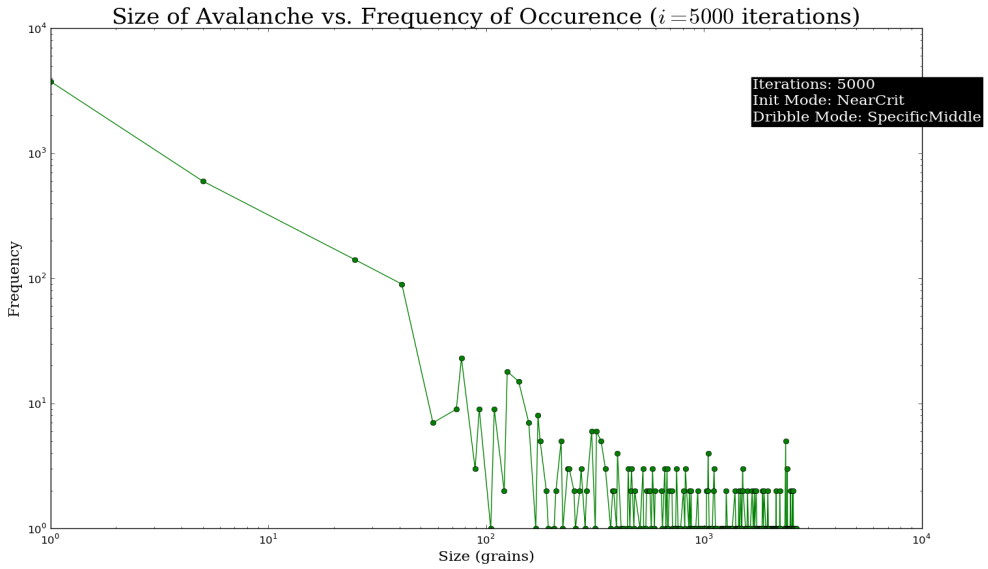

Figure 5: Size of Avalanche vs. Frequency of Size (Initialized at $s_{i,j} = 3$)

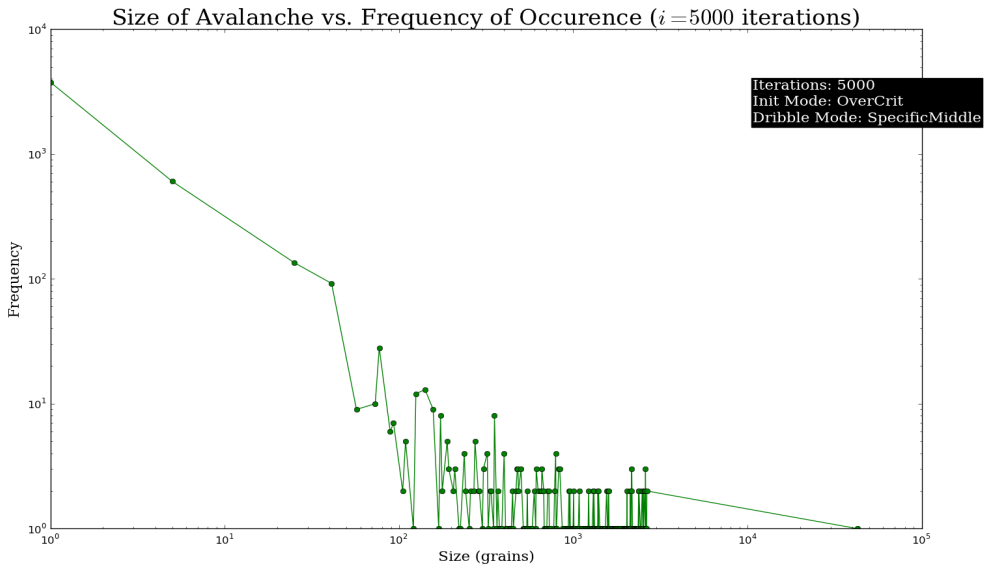

Figure 6: Size of Avalanche vs. Frequency of Size (Initialized at $s_{i,j} = 7$)

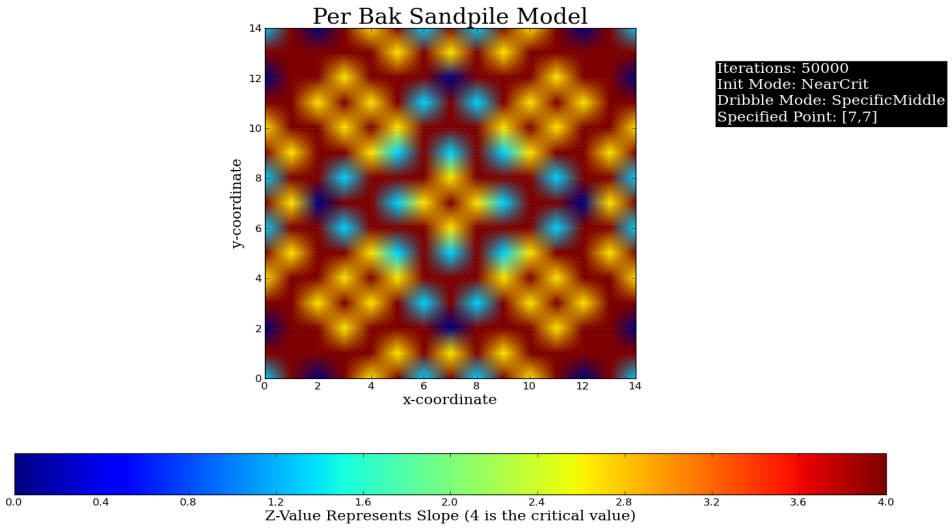

Figure 7: State of sandpile after 50000 middle grain drops (Initialized at $s_{i,j} = 3$)

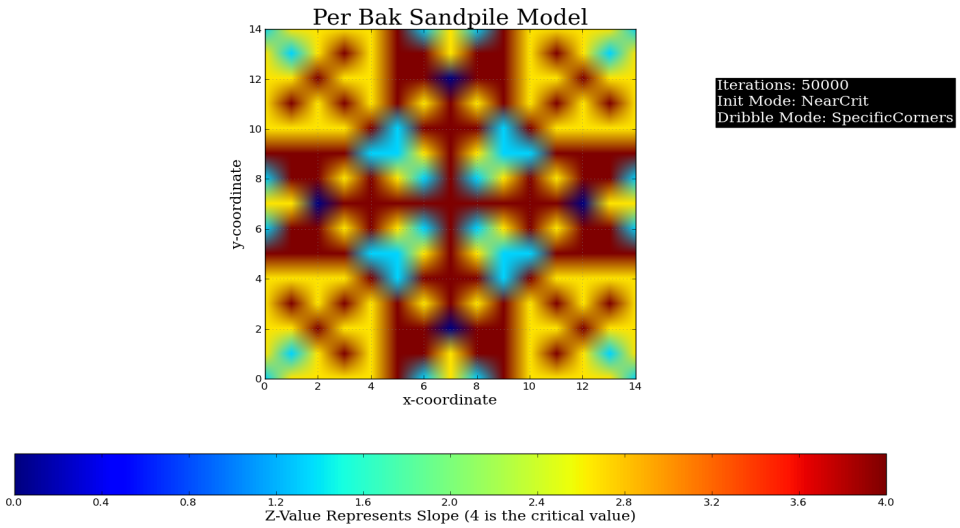

Figure 8: State of sandpile after 50000 corner grain drops (Initialized at $s_{i,j} = 3$)

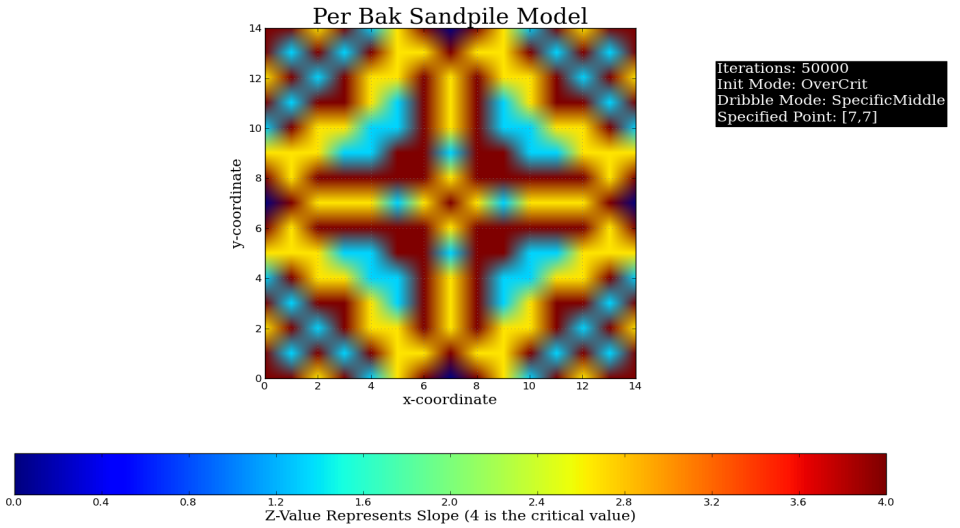

Figure 9: State of sandpile after 50000 middle grain drops (Initialized at $s_{i,j} = 7$)

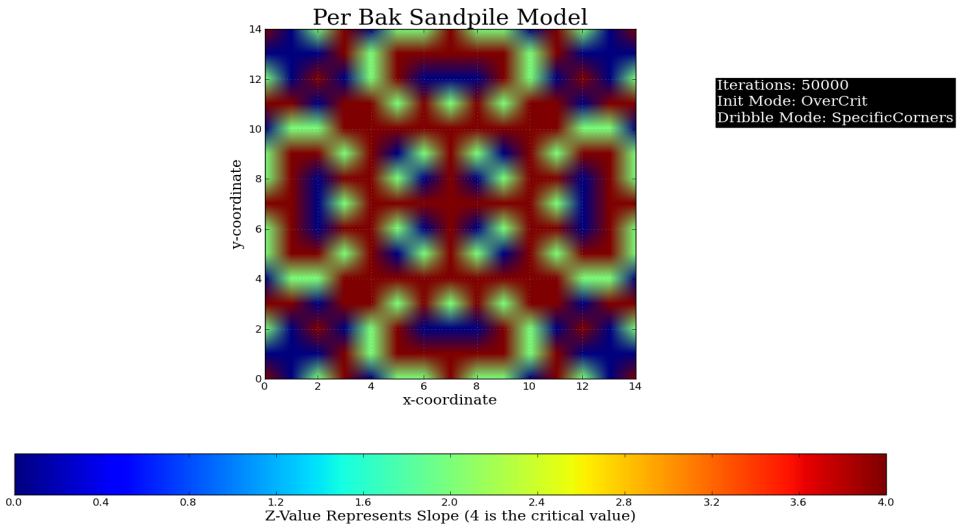

Figure 10: State of sandpile after 50000 corner grain drops (Initialized at $s_{i,j} = 7$)

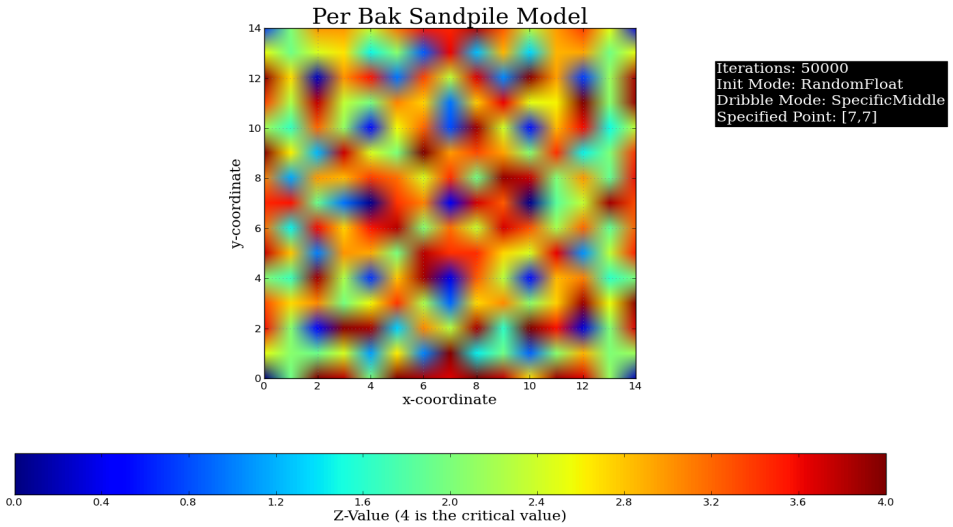

Figure 11: State of sandpile after 50000 middle grain drops (Initialized at $s_{i,j} = randFloat(0, 1)$)

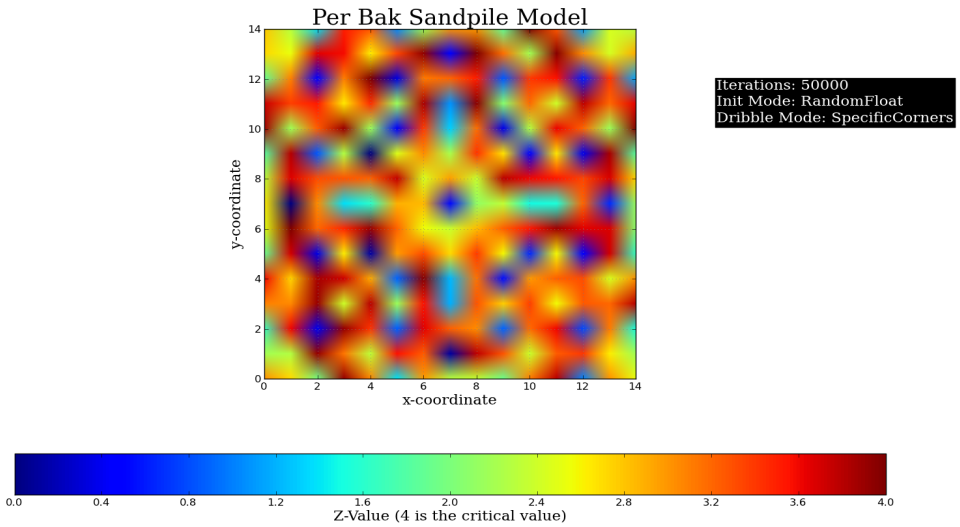

Figure 12: State of sandpile after 50000 corner grain drops (Initialized at $s_{i,j} = randFloat(0, 1)$)

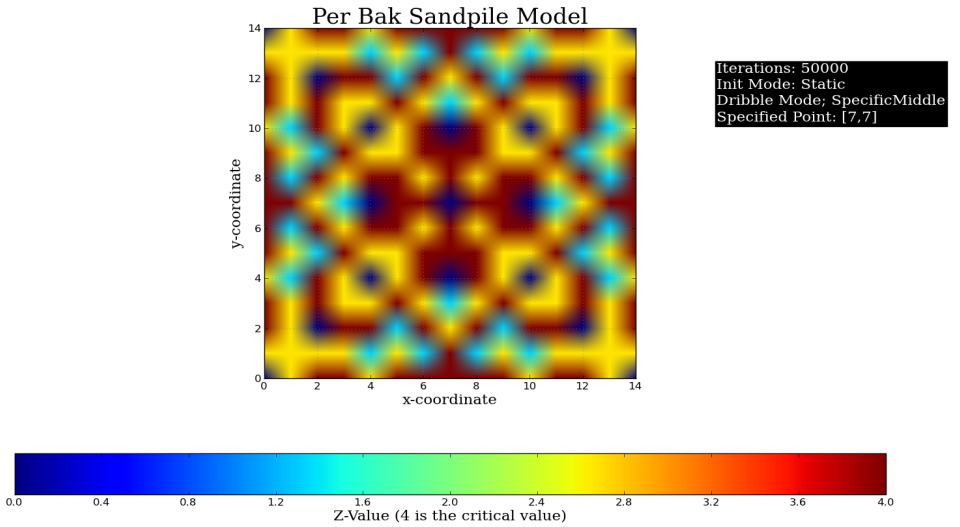

Figure 13: State of sandpile after 50000 middle grain drops (Initialized at $s_{i,j} = 0$)

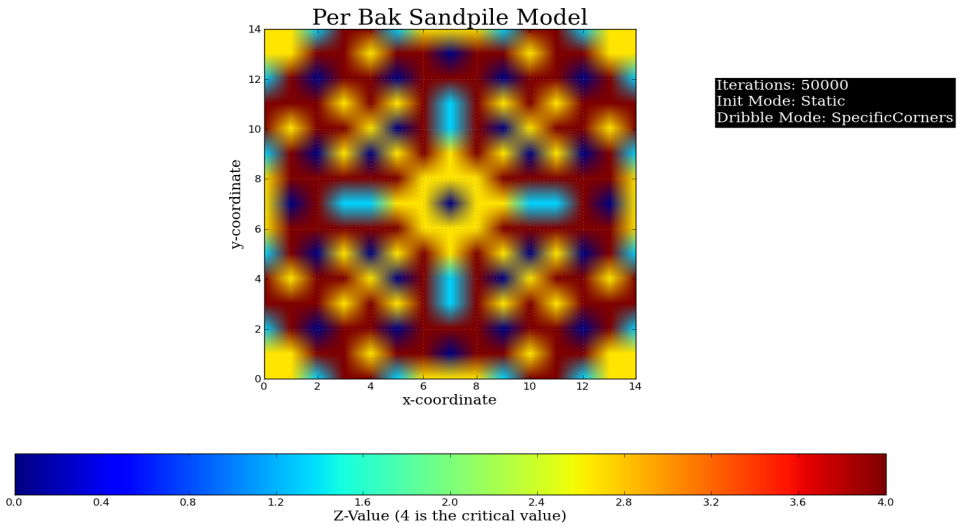

Figure 14: State of sandpile after 50000 corner grain drops (Initialized at $s_{i,j} = 0$)

4 Analysis

Figures 7-14 show several different initialization and dribble modes after $i = 50000$ iterations. Each lattice shows how the perturbations from the initial grain drop has cascaded after 50000 drops. The obvious result from a statically initialized sandpile will be symmetry, but it is interesting to see that even a randomly initialized pile eventually still maintains that symmetry. Also interesting is the fact that all seemed to follow the power law avalanche frequency rate.

As before, this model can also be shown to be extensible to an earthquake fault model (called the Bak-Tang Earthquake Fault Model) [4]. In this model, the Earth's crust is a periodic 2D lattice. There are blocks at each site, the lattice is connected with springs, and static/dynamic friction (stick-slip motion) exists. Blocks are pulled in a fixed direction by force due to tectonic plate motion. The 2D square lattice has side L and $N = L^2$ sites. A variable $z(i, j)$ represents force on block at site (i, j). Slip occurs at a critical threshold force z_{cr} (such as the critical value $z_{cr} = 4$ from the sandpile simulation). Initialize at time $t = 0$ and assign small random values to $z(i, j)$.

The loop is as follows:

Loop:

1. Increase each $z(i, j)$ by small p value (e.g., $p = .00001$ - [update value], and let $t \to t + 1$
2. If all $z(i, j) \leq z_{cr}$, fault is stable, go back to "Loop".
3. While any site (i, j) is unstable, decrease force by $z_{cr} = 4$ at (i, j) and increase force at each neighbor by $\frac{z_{cr}}{4} = 1$
4. The size of the event s is the total number of blocks that become unstable and slide.
5. The important take-away message is that an instability can trigger a very large event! [4]

5 Interpretation

This really does seem like compelling evidence for the existence of this "self-organized criticality". Everyday in nature, these complex critical states can be observed. Anything with that self-similar fractal-like structure can probably be traced back to this idea of complex physical structures being formed by the building up and toppling over based on states of criticality. In other words, structures are built up by the introduction of some new "grain of sand" onto the "sandpile". This grain could be introduced by the wind, rain, or a single footstep creating pressure and relieving sand from a higher elevation onto the pile. If this introduction puts the slope over the critical point, massive toppling occurs which in turn causes the newly created slopes to topple.

In the simplest terms, a slope wouldn't exist if it was beyond this "angle of repose", therefore everything currently observable sits within this angle or quickly adjusts until all its neighbors are within that angle (avalanches). This author is quite interested in learning whether this can be applied to the universe as a whole. Are galaxies formed in a similar fashion? That will be the subject of further research.

6 Critique

This project was incredibly insightful. The universe is full of symmetry and fractal self-similar behavior, so it comes as no surprise that BTW's demonstration shows how many of these structures could have been formed. If there had been more time, this project would have liked to be more interactive beyond simply changing parameters and viewing the results. It would be interesting to see how grains would shift if perturbated by a mouse click or drag.

It would have also been interesting to implement the Bak-Tang Earthquake Fault Model as a 2D cellular automaton. This could also be the subject of further research in this area.

References

[1] Per bak, Chao Tang, and Kurt Wiesenfeld, (March 13, 1987)
Self-Organized Criticality: An Explanation of 1/f Noise
Physics Department, Brookhaven National Laboratory, Upton New York 11973
Physical Review Letters Volume 59, Number 4 (July 27, 1987)

[2] Professor Leight Tesfatsion
Cellular Automata: Basic Intro
Economics Dept. Iowa State University
`http://www2.econ.iastate.edu/classes/econ308/tesfatsion/cellularautomataintro.lt.pdf`
Retrieved May 3rd. 2012

[3] Dr. Richard J. Gonsalves, Wed. April 5, 2010
Interdisciplinary Problems and Python Scripting
Lecture 5-3: Sandpile Automaton in 2-D
Computational Physics 2
`http://www.physics.buffalo.edu/phy411-506/topic5/lec-5-3.pdf`
Retrieved May 1st. 2012

[4] Dr. Richard J. Gonsalves, Wed. April 7, 2010
Interdisciplinary Problems and Python Scripting
Lecture 5-4: Earthquake Dynamics
Computational Physics 2
`http://www.physics.buffalo.edu/phy411-506/topic5/lec-5-4.pdf`
Retrieved May 1st. 2012

[5] The Bak-Tang-Wiesenfeld (BTW) sandpile applet
`http://www.cmth.bnl.gov/~maslov/Sandpile.htm`
Retrieved May 1st, 2012

[6] PyOpenGL for OpenGL Programmers
`http://pyopengl.sourceforge.net/documentation/opengl_diffs.html`
Retrieved May 7th, 2012