**PARIS 1 PANTHÉON-SORBONNE UNIVERSITY**

MASTER'S IN ECONOMETRICS AND STATISTICS

# Modeling Self-Organized Criticality

## The Bak–Tang–Wiesenfeld Sandpile Model

**Supervised by: Pr. Philippe de Peretti**

**Prepared by:**

Aya EL FARFAR
Aya WIFAK
Alicia DAHMANI

Academic Year 2025–2026

# Contents

# 1 Introduction

Per Bak, Chao Tang, and Kurt Wiesenfeld (1987) introduced Self-Organized Criticality (SOC) as a dynamical mechanism through which extended, slowly driven, locally interacting systems spontaneously evolve toward a critical state, without external parameter tuning. Using the sandpile cellular automaton, they demonstrated that once the system reaches this state, activity unfolds through avalanches of all sizes and durations, following power-law statistics for avalanche size $S$ and duration $T$, and exhibiting a $1/f$-like power spectrum in the dissipation time series $F(t)$.
SOC thus unifies two ubiquitous empirical regularities, fractal spatial structure and $1/f$ temporal noise, under a single organizing principle.
The objectives of this work are threefold:

1. To synthesize the BTW argument and clarify the interpretation of $1/f$ behavior as a power-law spectrum rather than an exponential decay.

2. To formalize SOC as an endogenous alternation between metastable accumulation and rapid avalanche phases, grounded in Abelian geometry on graphs.

3. To implement and analyze two complementary driving schemes, single-vertex and uniform random, in order to estimate avalanche size ($N$), duration ($T$), and the activity power spectral density $S(f)$.

# 2 BTW (1987): core result, implications and applications

## 2.1 Claim

Systems with many degrees of freedom that are weakly dissipative, locally interacting, and slowly driven can spontaneously form into a critical state characterised by a scale free activity. Avalanches in the Bak–Tang–Wiesenfeld (BTW) sandpile model exhibit power-law size and duration distributions, whereas the activity or dissipation time series shows a spectrum of the $1/f$ type. Thus, flicker noise in time and fractal spatial organisation are both explained by the same underlying mechanism.

## 2.2 Model

On a finite graph $G = (V, E)$ (e.g., an $L \times L$ square lattice), each site $i \in V$ holds an integer height $z_i \geq 0$. A slow drive adds one grain at a site. If $z_i$ reaches the threshold (on the square lattice, $z_c = 4$), site $i$ topples:

$$z_i \leftarrow z_i - 4, \qquad z_j \leftarrow z_j + 1 \;\; \forall\, j \sim i,$$

that is, a local application of the combinatorial Laplacian $\Delta$. Interior dynamics are conservative, while dissipation occurs only through open boundaries. Repeated drive–relaxation cycles bring the system to a stationary regime characterized by intermittent avalanche activity.

## 2.3 Specified implications

### 2.3.1 Endogenous criticality and finite-size scaling

Under the joint effects of slow driving, local threshold rules, interior conservation, and boundary dissipation, the sandpile self-organizes into a critical regime. In this regime, the distributions of avalanche size and duration follow power laws: $D(s) \sim s^{-\tau}$ and $D(T) \sim T^{-\alpha}$. The largest observable events are limited by geometry: the upper cutoffs $s_c$ and $T_c$ grow with system size and depend on boundary shape. When data are rescaled by these cutoffs, curves obtained at different sizes collapse onto a common form, confirming finite-size scaling. Changing which sides are open or the lattice dimension modifies cutoffs and critical exponents but preserves the algebraic character of the distributions.

### 2.3.2 Abelian/Laplacian control of relaxation

In the Bak–Tang–Wiesenfeld (BTW) sandpile model, the toppling dynamics are governed by the graph Laplacian: each unstable vertex loses a number of grains equal to its degree, while each neighbor gains one. This local and linear redistribution ensures that relaxation is Abelian, meaning that the final stable configuration is independent of the sequence of topplings.

With a designated sink, the set of recurrent configurations forms a finite Abelian group, known as the sandpile group, algebraically equivalent to the set of spanning trees of the underlying graph. The Green's function of the reduced Laplacian determines the mean avalanche footprint, while the spectral gap controls the relaxation timescale—smaller gaps correspond to longer avalanches and slower decay. Hence, the geometry and connectivity of the graph govern both the spatial and temporal properties of relaxation.

**The Abelian Property.** A group is Abelian if its internal operation is commutative, that is,

$$x * y = y * x \quad \text{for all } x, y \in G.$$

In the sandpile model, toppling operations commute: regardless of the order in which unstable sites are relaxed, the system converges to the same stable configuration. This commutativity constitutes the core of the model's Abelian structure, ensuring deterministic convergence within a probabilistic framework.

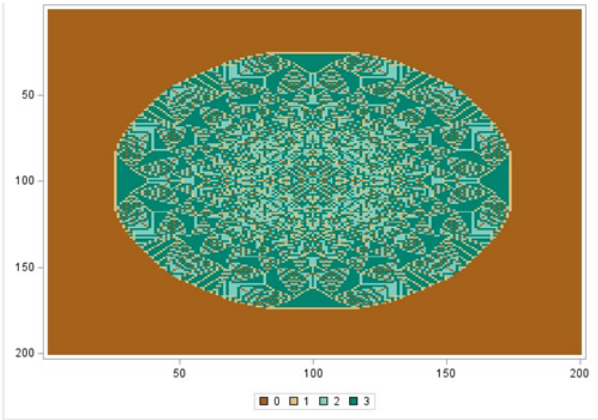



Figure 1: Abelian Sandpile: Grid 200x200 | 40000 grains

**Geometry and Interpretation.** Numerical simulations of the sandpile model produce symmetric or fractal spatial patterns, where each color represents the number of remaining grains per site. These patterns embody the system's Abelian geometry: despite the apparent local randomness of avalanches, the global configuration exhibits precise mathematical order. The Abelian structure explains the convergence to stability (internal organization), while the fractal morphology describes how this order manifests at macroscopic scales.

### 2.3.3 Temporal long memory from burst superposition

Because avalanches have heavy-tailed sizes and durations, their superposition in time produces a power-law spectrum $S(f) \sim f^{-\beta}$ (typically with $\beta \approx 1$). As a result, the activity is irregular and dispersed across various temporal scales, favouring risk declarations based on distribution and condition over speculative predictions. If the temporal scales are not distinguished—for instance, the drive is not slow—queues become erratic and the spectrum becomes white, which reduces or suppresses the regime 1/f β.

### 2.3.4 Robustness to disorder

Introducing moderate quenched disorder, such as bond removal, does not destroy criticality. The main effects are small shifts in cutoffs and amplitudes, while exponents and scaling forms remain stable. This robustness justifies focusing on scaling exponents and finite-size collapse rather than on microscopic details.

## 2.4 Applications

### 2.4.1 Forest fires

Forest fires are another example of self-organized criticality in nature. In a forest, vegetation grows slowly, increasing the amount of fuel available. Occasionally, a spark, from lightning or human activity, ignites a fire. The fire then spreads locally from tree to tree. As forests regrow, the system self-organizes toward a critical density where small and large

fires coexist.
The distribution of burned areas follows a power law, meaning there is no typical fire size, small fires are frequent, large ones rare but statistically related. This reflects the same balance of slow accumulation (tree growth) and sudden release (fire) seen in sandpile avalanches.
SOC thus explains the scale-free patterns and intermittency observed in real fire data, showing how local interactions and gradual energy buildup naturally lead to complex, unpredictable events without external tuning.

### 2.4.2 Neuronal avalanches (cortical networks)

In the brain, neurons do not fire randomly but often activate in bursts called neuronal avalanches. These cascades vary greatly in size, some involve only a few neurons, others spread across large parts of the network.
Beggs and Plenz (2003) discovered that the size of these avalanches follows a power-law distribution with an exponent close to $-3/2$, the same pattern predicted by the critical sandpile model. This means there is no typical event size: small and large avalanches coexist.
This finding suggests that cortical networks naturally evolve toward a critical state, a balance between silence and chaos, where activity can spread efficiently without destabilizing the system. In this regime, the brain achieves optimal information transmission, flexible responses, and long-range correlations over time, characteristic of $1/f$-type temporal noise.

# 3 Exponential decay and power law behavior "1/f"

## 3.1 Exponential decay (distribution with a scale)

In many physical and stochastic systems, event sizes or waiting times follow exponential decay, meaning that large events become exponentially rare:

$$p(x) = \frac{1}{x_0}\, e^{-x/x_0},\ \ x \geq 0.$$

Here, $\lambda$ is a characteristic rate or scale. This kind of distribution suggests that correlations decay quickly over time and that the system has a typical scale, such as a typical avalanche size, earthquake magnitude, or energy release. This is how most non-critical systems operate. Exponentials show up on plots as straight lines on semi-log axes.

### 3.2 Power-law (scale-free) statistics

Self-organized critical systems display power-law statistics, where:

$$D(s) \propto s^{-\tau}, \quad D(T) \propto T^{-\alpha}, \quad \langle s \rangle(T) \propto T^{\gamma}.$$

A power law has no intrinsic scale. Small and large events belong to the same statistical hierarchy, reflecting scale invariance. In the frequency domain, this corresponds to a $1/f^{-\beta}$ power spectrum, known as a flicker noise or pink noise:

$$S(f) \propto f^{-\beta} \quad (0 < \beta < 2, \text{ often } \beta \approx 1).$$

This slow, algebraic decay indicates long-range temporal correlations: fluctuations persist across many time scales, unlike the rapid damping of exponential systems. Hence, while exponential decay characterizes systems with short memory and finite correlation lengths, power laws ($1/f$ spectra) signal criticality, self-similarity, and long-memory dynamics, the statistical signature of self-organized criticality.

## 4 Formal definition of SOC and endogenous phase transitions

In the Bak–Tang–Wiesenfeld framework, a system is said to exhibit Self-Organized Criticality (SOC) when it autonomously evolves toward a critical attractor, where fluctuations span all scales and macroscopic observables follow power-law statistics. No external control parameter is tuned, the critical point is reached endogenously through the system's own dynamics.

The process alternates between two intrinsic regimes, a slow accumulation phase, during which the drive adds energy or mass to the system, and a fast avalanche phase, where local instabilities trigger cascades that dissipate the excess.

Through this continual alternation, the system stabilizes around a self-regulated critical state: small perturbations can trigger events of any size, but the average input and dissipation remain balanced over time.

Formally, the stationary dynamics satisfy:

$$P(s) \sim s^{-\tau}, \qquad P(T) \sim T^{-\alpha},$$

where $s$ and $T$ denote the size and duration of the avalanche, and the exponents $\tau$ and $\alpha$ reflect the universality class of the underlying graph or geometry.

This endogenous phase transition differs from classical criticality: instead of being externally tuned (as in temperature-driven phase changes), the system's own feedback between slow loading and fast relaxation continuously drives it to the critical threshold where scale invariance emerges.

# 5 Modeling I — Center-driven model

## 5.1 The intuitive version

### 5.1.1 Algorithmic logic

In this section, we implement the sandpile logic on a matrix.
We start with a matrix ($n \times n$) prefilled with zeros. Each square (cell) in the matrix can contain a number of grains.
At each iteration, a grain of sand is placed in the square at the center of the matrix. When a cell contains 4 grains, it becomes unstable and loses all its grains; each of these grains is given to one of its four neighboring cells (if they exist).
This redistribution can make other cells unstable, leading to a series of avalanches until the system is fully stabilized, i.e., until all cells contain fewer than 4 grains.

### 5.1.2 Modelling under SAS IML

```
proc iml;
/* Create a 50x50 matrix of zeros */
Tas = j(50, 50, 0);
n = nrow(Tas); /* number of rows = 50 */
centre = ceil(n/2);

/* Loop: drop Ngrains one by one */
do Ngrains = 1 to 4000;
   Tas[centre, centre] = Tas[centre, centre] + 1;

   /* 3. Topple all cells >= 4 */
   repeat = 1;
   do while (repeat);
      repeat = 0; /* assume no more topplings */

      do r = 1 to 50;
         do c = 1 to 50;

            /* If a cell has 4 grains or more : it topples */
            if Tas[r,c] >= 4 then do;
               Tas[r,c] = Tas[r,c] - 4;

               /* Give +1 to neighbouring cells if they exist */
               if r > 1  then Tas[r-1,c] = Tas[r-1,c] + 1; /* UP */
               if r < 50 then Tas[r+1,c] = Tas[r+1,c] + 1; /* DOWN */
               if c > 1  then Tas[r,c-1] = Tas[r,c-1] + 1; /* LEFT */
               if c < 50 then Tas[r,c+1] = Tas[r,c+1] + 1; /* RIGHT */
```

```
                /* Reset repeat=1 since some cells may now topple */
                repeat = 1;
            end;
        end;
    end;
  end;
end;

print tas;

call heatmapdisc(Tas)
  xvalues = 1:n
  yvalues = 1:n
  title = "Sandpile 50x50 4000 grains";
quit;
```

Listing 1: SAS IML implementation of the intuitive version

However, this program contains several loops, which makes its execution quite slow. For this reason, the use of an optimized version is preferred for larger matrices and a higher number of grains.

## 5.2 The optimised version

### 5.2.1 Algorithmic logic

The optimized version of this program retains the same structure and logic as the previous version. Significant changes have been added :

- Adding grains: Instead of adding N grains one by one, they are all deposited at once in the center, eliminating the loop that made successive deposits.
- Stabilization: The `loc()` function is used to find unstable cells. The grains to be redistributed are temporarily stored in an auxiliary matrix called `Add`. This matrix is then added to the main matrix all at once, which significantly reduces the number of loops.

### 5.2.2 Modelling under SAS IML

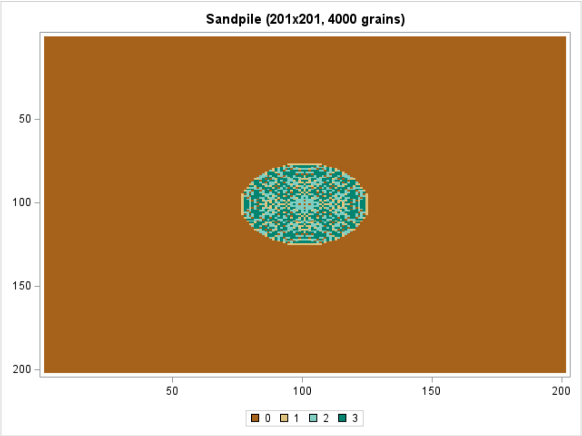
```
proc iml;
/* Model parameters */
n = 201;
centre = ceil(n/2);

Tas = j(n, n, 0);
Tas[centre, centre] = 400000;
```

```
/* Critical threshold */
do while (max(Tas) >= 4);
   idx = loc(Tas >= 4);                 /* locate the cells to topple */
   if ncol(idx)=0 then leave;

   /* Positions (r,c) of these cells */
   rr = ceil(idx / n);
   cc = idx - (rr - 1)*n; *or "cc = mod(idx - 1, n) + 1";

   /* Remove 4 grains simultaneously */
   do k = 1 to ncol(idx);
      Tas[ rr[k], cc[k] ] = Tas[ rr[k], cc[k] ] - 4;
   end;

   /* Accumulate redistributed grains */
   Add = j(n,n,0); *Add stores the grains redistributed to neighbours;

   do k = 1 to ncol(idx);
      r = rr[k];  c = cc[k];
      if r>1  then Add[r-1,c] = Add[r-1,c] + 1;  /* up   */
      if r<n  then Add[r+1,c] = Add[r+1,c] + 1;  /* down */
      if c>1  then Add[r,c-1] = Add[r,c-1] + 1;  /* left */
      if c<n  then Add[r,c+1] = Add[r,c+1] + 1;  /* right */
   end;

   Tas = Tas + Add;   /* update the grid */
end;

/* Visualisation: Heatmap */
Sandpile = mod(Tas, 4);
call heatmapdisc(Sandpile)
   xvalues=1:n  yvalues=1:n
   title="Abelian sandpile (201x201, 400000 grains)"
   displayoutlines=0;
quit;
```

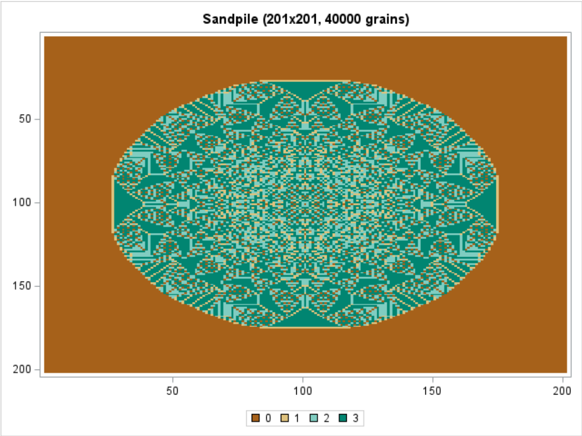
Listing 2: SAS IML implementation of the optimised version

## 5.3 Results on SAS IML

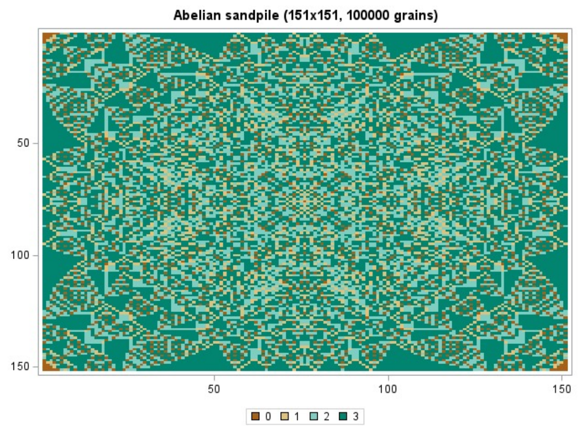
The following illustrations representing the SandPile configuration were obtained for several grid sizes and for different numbers of deposited grains.



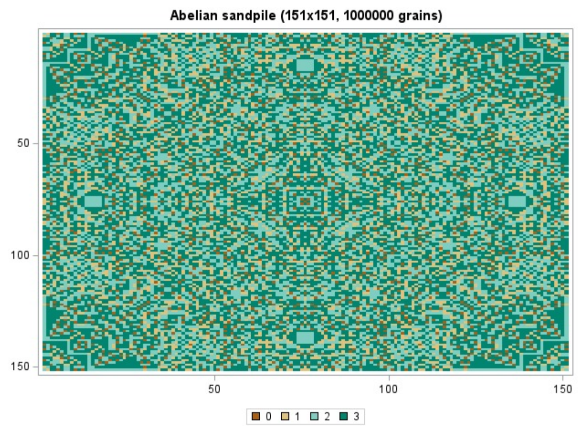

(a) Grid 201x201 | 4000 grains



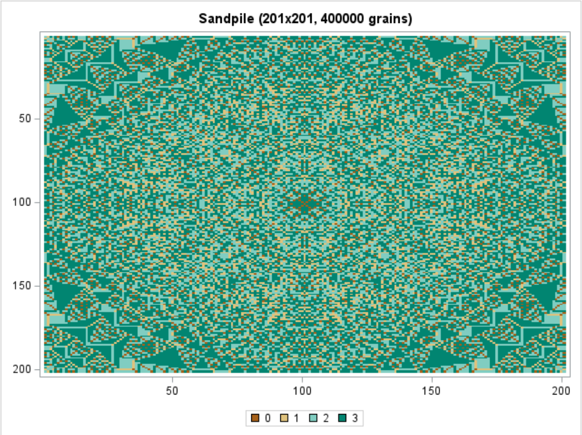

(b) Grid 201x201 | 40000 grains



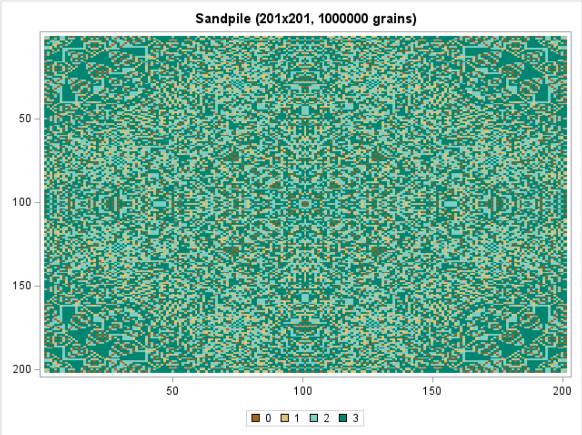

(c) Grid 151x151 | 100000 grains




(d) Grid 151x151 | 1000000 grains




(e) Grid 201x201 | 400000 grains




(f) Grid 201x201 | 1000000 grains

Figure 2: Final configurations of the Abelian sandpile model for various grid sizes and grain counts.

## 5.4 Corner-driven variant

In this variant of the model, the same logic is retained, except for the placement of the grains. At each iteration, one grain is placed simultaneously at each of the four corners of the matrix.

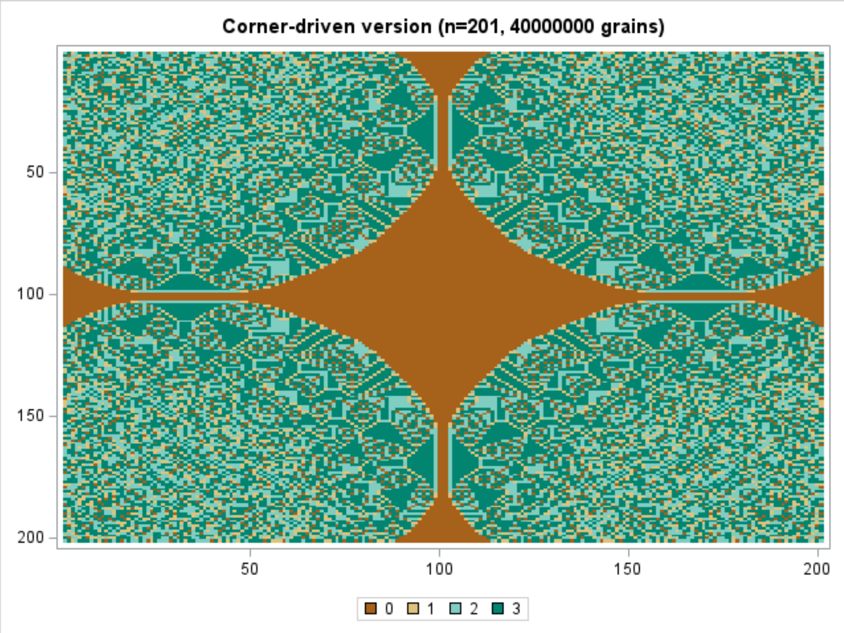



Figure 3: Corner-driven: Grid 201x201 | 40000000 grains

# 6 Modeling II — Random-driven model and 1/f spectrum

## 6.1 Random SandPile code

```
proc iml;
   /*parameters*/
   n = 50;
   Ngrains = 200000;
   Tas = j(n, n, 0);

   /* storage vectors */
   Size = j(1, Ngrains, 0);            /* number of affected sites */
   Time  = j(1, Ngrains, 0);           /* duration (number of iterations
      before stability) */

   do i = 1 to Ngrains;

      /* random deposition */
      r = ceil(n * rand("Uniform"));
      c = ceil(n * rand("Uniform"));
      Tas[r, c] = Tas[r, c] + 1;

      /* stabilization */
      do while (max(Tas) >= 4);
         idx = loc(Tas >= 4);
         if ncol(idx) = 0 then leave;
```

```
    /* variable storage */
    Size[i] = Size[i] + ncol(idx);        /* number of avalanches */
    Time[i]  = Time[i] + 1;               /* avalanche duration */
       rr = ceil(idx / n);
       cc = idx - (rr - 1)*n;

       /* simultaneous removal */
       do k = 1 to ncol(idx);
          Tas[rr[k], cc[k]] = Tas[rr[k], cc[k]] - 4;
       end;

       /* simultaneous redistribution */
       Add = j(n, n, 0);
       do k = 1 to ncol(idx);
          r = rr[k]; c = cc[k];
          if r > 1  then Add[r-1, c] = Add[r-1, c] + 1;
          if r < n  then Add[r+1, c] = Add[r+1, c] + 1;
          if c > 1  then Add[r, c-1] = Add[r, c-1] + 1;
          if c < n  then Add[r, c+1] = Add[r, c+1] + 1;
       end;

       Tas = Tas + Add;               /* simultaneous update */
    end;

 end;

 /*final visualization*/
 Sandpile = mod(Tas, 4);
 call heatmapdisc(Sandpile)
      xvalues=1:n yvalues=1:n
      title="Abelian Sandpile (random)"
      displayoutlines=0;
```

Listing 3: Random Sandpile Code

## Code Description

The code above represents the random Abelian version of the Sandpile Model. It begins by initializing the matrix `Tas` and the two storage vectors `Size` and `Time`, which respectively record the size and duration of each avalanche throughout the simulation.

The **main loop** ensures that, for each deposited grain (from 1 to `Ngrains`), a grain is added at a random position $(r, c)$. After each addition, the system checks whether some cells contain more than four grains.

The **inner loop** handles the stabilization process. It identifies unstable cells using `loc(Tas >= 4)`, stores the avalanche's size and duration, and makes these cells topple into the auxiliary matrix Add, which is then added back to the main matrix Tas.

## 6.2 Creation of logarithmic variables and frequency tables

```
Size_pos = Size[loc(Size > 0)]; /* to filter out size = 0  */
call tabulate(valueS, countS, Size_pos);
frequencyS = countS/sum(countS);

log_freq_size = log(countS/sum(countS));
log_size = log(valueS);

Time_pos = Time[loc(Time > 0)]; /* to filter out initial Time = 0 */
call tabulate(valueT, countT, Time_pos);
frequencyT = countT/sum(countT);

/*export tables as datasets*/

/* for size */
create log_size_distribution var {"valueS" "frequencyS" "log_size" "log_
    freq_size"};
append;
close log_size_distribution;

/* for duration */
create log_time_distribution var {"valueT" "frequencyT"};
append;
close log_time_distribution;
```

Listing 4: Random Sandpile Code

### Code Description

In the first part of this code, the data are filtered to retain only the avalanches with non-zero size and duration. The function call tabulate() counts the number of occurrences of each unique value. Then, the relative frequencies are computed, and the data are transformed into their logarithmic form in order to perform the **log–log power-law analysis**. Finally, a dataset is created to export the results for further analysis and visualization.

## 6.3 Data visualization and power-law 1/f distribution

```
title "Frequency of avalanche sizes";
call histogram(Size) scale='PROPORTION';
```

```
title "Frequency of avalanche durations (number of iterations before
    stability)";
call histogram(Time) scale='PROPORTION';

quit;

proc sgplot data=log_size_distribution;
scatter x=valueS y=frequencyS / markerattrs = (symbol=circlefilled);
xaxis type=log label="Avalanche size" grid;
yaxis type=log label="Frequency of size" grid;
title "Log-log distribution of avalanche sizes (1/f noise)";
run;

proc sgplot data=log_time_distribution;
   scatter x=valueT y=frequencyT / markerattrs = (symbol=circlefilled);
   xaxis type=log label="Time" grid;
   yaxis type=log label="Frequency of time" grid;
   title "Log-log distribution of avalanche durations";
run;

/*Regression*/
proc reg data=log_size_distribution(where=(log_Size <= 5));
   model log_freq_size = log_size;
run;
```

Listing 5: Random Sandpile Code

# 7 Results and Discussion

This section presents and discusses the results produced by the SAS IML code introduced in the previous section.
We focus on the random–drive configuration, which allows for the observation of self–organized criticality emerging spontaneously from random grain deposition.
Through this combination of spatial and statistical perspectives, we aim to demonstrate how local threshold dynamics give rise to scale–free behavior and $1/f$ noise.

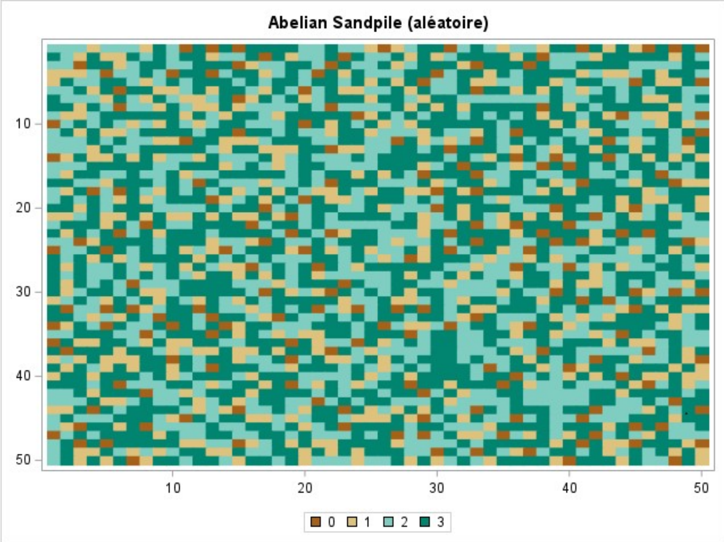
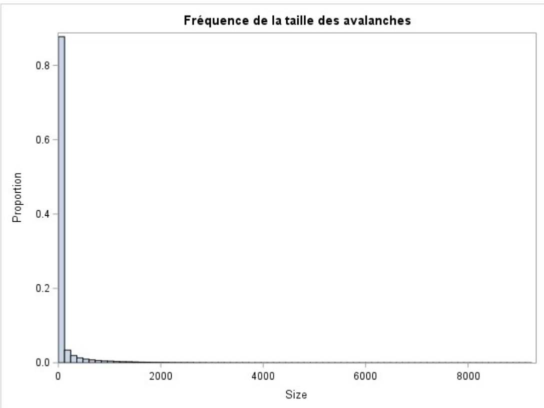
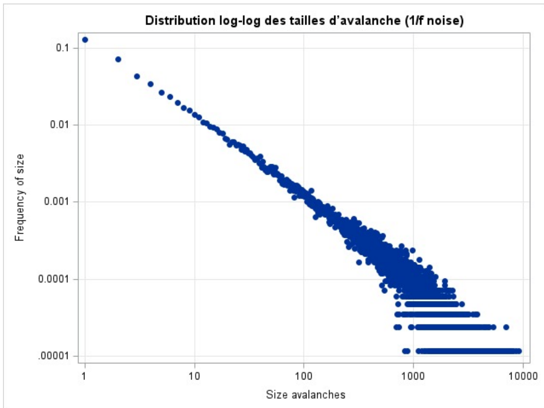



Figure 4: Final configuration of the Abelian sandpile (random drive $50 \times 50$ lattice, 200,000 iterations).

Figures 4 and 5 present the main results of the simulation, showing how the Abelian sandpile evolves over time through avalanches of different sizes and durations.

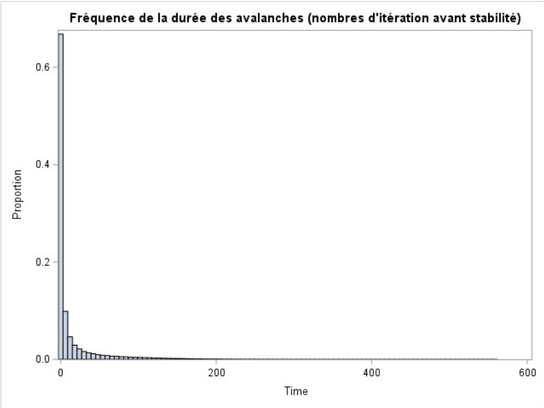



(a) Frequency of avalanche sizes.

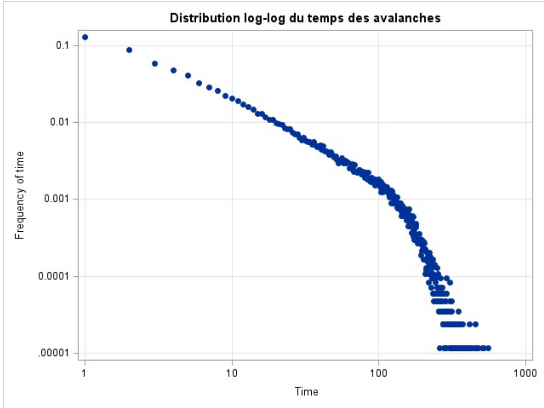



(b) Log–log distribution of avalanche sizes

Figure 5: Avalanche size distribution — $50 \times 50$ lattice, 200,000 iterations.




(a) Frequency of avalanche durations.




(b) Log–log distribution of avalanche durations.

Figure 6: Avalanche duration distribution — 50 × 50 lattice, 200,000 iterations

Panels (a) and (b) display the empirical distributions of avalanche sizes and durations, respectively. In both cases, most avalanches remain localised or short-lived, while large or long-lasting events are rare. On a log–log scale, the linear decay across several orders of magnitude reveals a clear power-law pattern,

$$P(s) \sim s^{-\tau}, \qquad P(T) \sim T^{-\alpha},$$

consistent with the signature of self-organized criticality.

To quantitatively confirm this scaling behavior, a log–log regression was performed on the avalanche size distribution.

La procédure REG
Modèle : MODEL1
Variable dépendante : log_freq_size

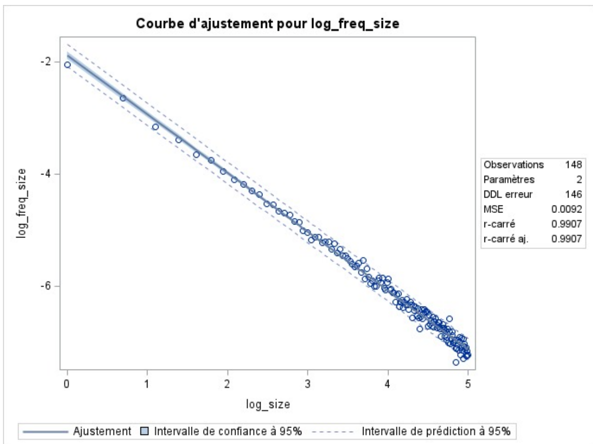
| | |
|---|---|
| Nb d'observations lues | 148 |
| Nb d'obs. utilisées | 148 |

| Analyse de variance | | | | | |
|---|---|---|---|---|---|
| Source | DDL | Somme des carrés | Moyenne quadratique | Valeur F | Pr > F |
| Modèle | 1 | 144.00014 | 144.00014 | 15632.8 | <.0001 |
| Erreur | 146 | 1.34486 | 0.00921 | | |
| Total sommes corrigées | 147 | 145.34500 | | | |

| | | | |
|---|---|---|---|
| Root MSE | 0.09598 | R carré | 0.9907 |
| Moyenne dépendante | -6.09667 | R car. ajust. | 0.9907 |
| Coeff Var | -1.57424 | | |

| Paramètres estimés | | | | | |
|---|---|---|---|---|---|
| Variable | DDL | Valeur estimée des paramètres | Erreur type | Valeur du test t | Pr > \|t\| |
| Intercept | 1 | -1.88289 | 0.03461 | -54.40 | <.0001 |
| log_size | 1 | -1.04812 | 0.00838 | -125.03 | <.0001 |

(a) SAS regression output for the log–log model




(b) Fitted regression curve with 95% confidence intervals

Figure 7: Regression analysis of avalanche size and frequency in log–log scale.

In summary, an ordinary least squares regression over the scaling window gives an estimated

slope of $\hat{\tau} = 1.048$, with $R^2 = 0.99$ and $p < 10^{-4}$,indicating that the regression is highly significant and explains 99% of the variance.
The slight curvature observed for large $s$ values (around $s \gtrsim s^*$) is likely due to finite-size effects, as the boundaries of the grid limit the spread of very large avalanches. The obtained exponent $\hat{\tau} \approx 1.04$ falls within the canonical range reported for the two–dimensional Bak–Tang–Wiesenfeld (BTW) sandpile model ($1 \lesssim \tau \lesssim 1.3$), confirming the presence of self–organized criticality under random drive.

Overall, these results confirm that the system reaches a stationary critical state characterized by scale-free avalanches (see Sections 3–4 for the theoretical background).

# 8 Extensions to earthquake modeling (slow–fast dynamics)

The analogy between sandpile avalanches and earthquakes has motivated both numerical and laboratory studies of fault dynamics as self-organized critical systems. In this framework, the slow accumulation of tectonic stress corresponds to the gradual addition of grains, and earthquake ruptures correspond to avalanches that release stored energy through local failures.
Yoshioka (2003) conducted a sandpile experiment using real sand deposited slowly onto a circular disk, observing how the ratio of grain size to disk size controls the system's behavior. Three distinct regimes were identified:

1. a SOC regime, where avalanche sizes obey the Gutenberg–Richter law (power-law frequency–magnitude distribution);

2. a Characteristic Earthquake (CE) regime, where large events occur quasi-periodically with nearly constant magnitude;

3. a transition regime between the two, observed when the ratio of grain diameter to disk radius approaches $\approx 0.02$.

This transition reflects a shift in the internal stress profile of the pile: small systems with more uniform stress behave as SOC systems, while larger or more rigid configurations develop stress concentration zones, leading to more regular, characteristic events.
The key insight is that SOC and periodic (CE) behavior coexist depending on the balance between slow loading and stress redistribution. In the context of fault systems, this corresponds to a slow–fast dynamical interaction:

1. slow tectonic loading drives the crust toward a critical state,

2. local ruptures release energy through avalanches,

3. and structural heterogeneity (geometry, friction, or coupling) can shift the system from stochastic SOC-like behavior to quasi-periodic CE-like cycles.

Thus, SOC provides a unifying framework linking the scale-free statistics of seismicity (Gutenberg–Richter law) with the emergence of periodic large earthquakes. Understanding this transition may help clarify why some faults behave critically while others repeat predictably, offering a bridge between statistical models of seismicity and deterministic earthquake cycles.

# 9 References


[1] Bak, P., Tang, C., & Wiesenfeld, K. (1987). *Self-organized criticality.* Physical Review Letters, 59(4), 381–384.

[2] Bak, P., Chen, K., & Tang, C. (1990). *A forest-fire model and some thoughts on turbulence.* Physics Letters A, 147(5–6), 297–300. https://doi.org/10.1016/0375-9601(90)90451-S

[3] Beggs, J. M., & Plenz, D. (2003). *Neuronal avalanches in neocortical circuits.* **Journal of Neuroscience, 23(35)**, 11167–11177.

[4] Chialvo, D. R. (2010). *Emergent complex neural dynamics.* Nature Physics, 6, 744–750. https://doi.org/10.1038/nphys1803