Simulating Self–Organised Criticality

Student Team Project
supervisor: prof. dr hab. Jakub Tworzydło

Dominik Stańczak, Oleg Kmechak, Grzegorz Skóra, and Aleksandra Jankowska

Department of Physics, University of Warsaw, Poland

February 28, 2020

Abstract

A Python framework for simulating Self–Organized Criticality on 2D grids is developed using typical best practices in a team software development. Manna model, Bak–Tang–Wiesenfeld sandpile model, Olami–Feder–Christensen earthquake model and Bak–Chen–Tang forest fire model were implemented. The critical exponents were gathered from the log–log histograms of the avalanche sizes from the simulations and they show good agreement with previous results of the simulations from literature.

1 Introduction

A motivation of this project is the lack of a simple educational tool for performing simulations of the Self–Organized Criticality models in the Python ecosystem. The version we were able to arrive at fulfills this purpose and allows further exploration of this research area.

The idea of the Self–Organized Criticality was introduced to grasp the feature of many systems, which is evolution to a critical state ([1, p. 6690]). The system can move away ("avalanche") from this equilibrium, but then it evolves back.

One of the observables is a size of the avalanche, defined as the number of places (grid nodes) which have been changed during the avalanche. One value of this avalanche size is gathered during the simulation for each drive–topple–dissipate cycle. From the histogram of avalanche sizes in log–log space, one can parse the "apparent exponent" ([2, p. 32]) by fitting a line to the log–log–linear part of the plot.

1.1 Models

1.1.1 The Bak–Tang–Wiesenfeld Sandpile Model

The model was first published in [3] and is motivated by the avalanching behaviour of a real sandpile. It can be defined in any number of dimensions, but is solved exactly in the 1D case. In its original definition the BTW Sandpile model is non–abelian. The model is considered deterministic due to the character of its toppling rule.

Rules [2, p. 86]

- d-dimensional regular lattice.
- Choose critical slope $z^c = 2d - 1$.
- Each site has slope z_n.
- Add a grain at n_0 chosen at random and update all uphill nearest neighbours n_0' of n_0:
  $z_{n_0} \to z_{n_0} + d$
  $z_{n_0'} \to z_{n_0'} - 1$.
- For each site n with $z_n > z^c$ distribute $2d$ grains among its nearest neighbours n'
  $z_n \to z_n - 2d$
  $\forall_{n'}\ z_{n'} \to z_{n'} + 1$.

- Grains are lost at open boundaries.

1.1.2 The Manna Model

The model was first published in [4]. It is a stochastic (due to its stochastic relaxation), 'two–state version' of the BTW Model. It can be defined in any dimension, but is non–trivial even in one. In its original definition the model is non–abelian, the abelian model was presented by Dhar in [5].

Rules for the non–abelian version [2, p. 164]

- d-dimensional regular lattice.
- Each site contains h_n particles.
- Add a particle at randomly chosen site n_0, $h_{n_0} \to h_{n_0} + 1$.
- For each site n with $h_n > h^c = 1$ distribute all particles randomly and independently among its nearest neighbours n', $h_{n'} \to h_{n'} + 1$ (h_n times) and reset $h_n \to 0$.
- Grains are lost at open boundaries.

1.1.3 The Olami–Feder–Christensen Model

The model described in [6] is based on the Burridge–Knopoff model of earthquake dynamics. While it is defined in any number of dimensions, the extensively studied case is 2D. The OFC model is deterministic and non–abelian.

Rules [2, p. 127]

- d-dimensional regular lattice with open boundaries.
- A force F_n acts on each site n.
- Conservation level $\alpha \in [0, 1/2d]$.
- Global threshold F_{th} usually unity.
- F_n taken randomly and independently from a uniform distribution in $[0, F_{th}]$.

- For each site n where $F_n \geq F_{th}$, move αF_n to the nearest neighbours n' to n
  $F_{n'} \rightarrow F_{n'} + \alpha F_n$
  and reset n: $F_n \rightarrow 0$.
- Energy is lost at toppling for $\alpha < 1/q$ and at open boundaries.

1.1.4 The Bak–Chen–Tang Forest Fire Model

The model proposed in [7] is motivated by fractal dissipation events in turbulence and linked to percolation. It is a completely different model compared with the sandpile–like models, as there is no flux of any locally conserved quantity. Additionally its driving is stochastic by definition. In 2D case it to be not critical.

Rules [2, p. 113]

- d-dimensional regular lattice.
- Every site is in one of the three states: Ash, Tree or Burning.
- Discrete time t (updates in sweeps).
- A site containing Ash at time t becomes occupied by a Tree at time $t + 1$ with small probability p.
- A site occupied by a Tree at time t and with nearest neighbour burning at time t starts Burning at time $t + 1$.
- A site Burning at time t contains Ash at time $t + 1$.

Moreover, in our simulations a site occupied by a Tree at time t can be "struck by lightning" with a small probability f and start Burning at time $t + 1$. This rule is not included in the original model.

2 Implementation

Python was chosen for the implementation due to being simple, readable and well known in the physics community. It can also be optimized where necessary. We used numpy arrays[8] as our basic data structure for the simulation grids, and the element-wise algorithms were implemented using the Numba[9] just-in-time compilation library. Data analysis was carried out mostly using the pandas library[10].

We also experimented with the Dask[11] parallel computing library to gather more samples.

The code is available on the GitHub platform: https://github.com/SocSIM/SocSIM

Examples of use and documentation can be found at: https://socsim.readthedocs.io

3 Simulations

The histograms of the data collected from the exemplary simulations for the examined models are shown on the Figs. 1-3. The exponents for the Manna model and BTW model are 1.06 and 1.38, respectively. The values of the exponents for the OFC model with different conservation levels are shown on the Fig. 4. Gathering an exponent was not applicable to the forest fire model implemented in our code.

4 Discussion

The log–log–linear fragment of histograms corresponds to power law scaling, which is one of the signs of Self–Organized Criticality.

There is a range of values of the exponents for a specific model. For example, for the 2D abelian Manna model, different authors give values between 1.253 and 1.30 ([2, p. 168]), for 2D BTW model – between 1 and 1.367 ([12, p. 10]), whereas for the OFC model, the exponent depends on the conservation level as shown in Fig. 4. The examples given above show, that our simulations result in exponent values close to those in the literature.

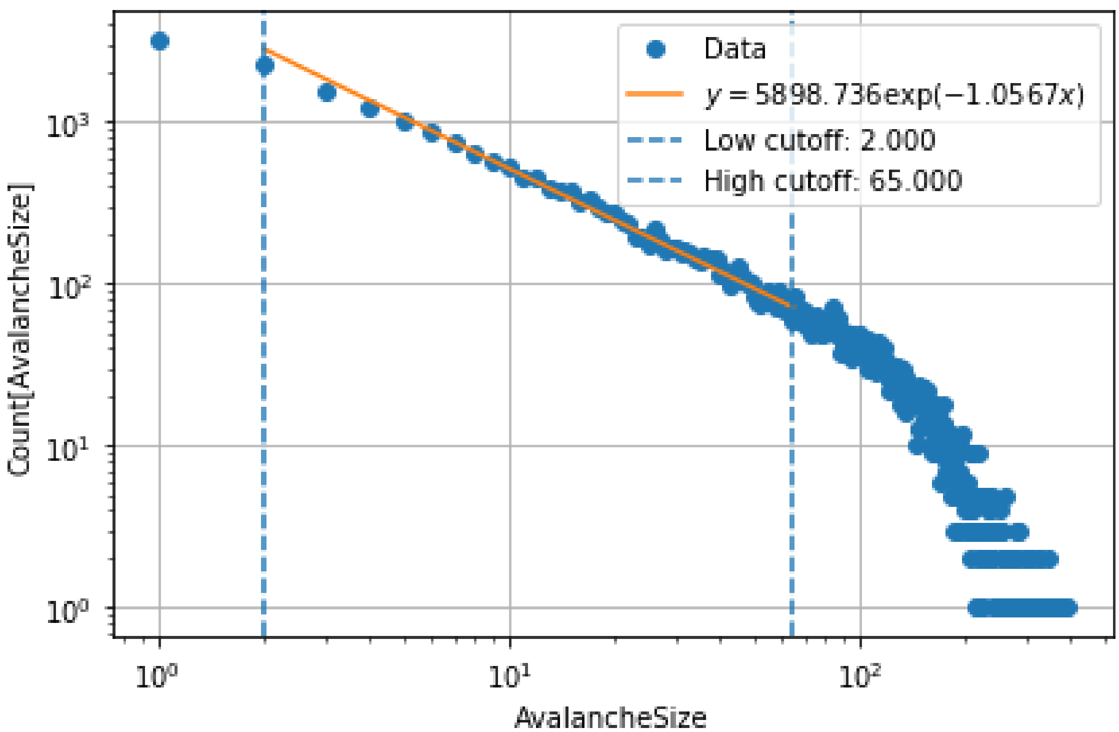

Figure 1: Number of the avalanches with the size given on the abscissa for the BTW model with the linear grid size $L = 50$, number of iterations $N = 6 \cdot 10^4$ counting after an initial $3 \cdot 10^4$ ("thermalization surplus") iterations.

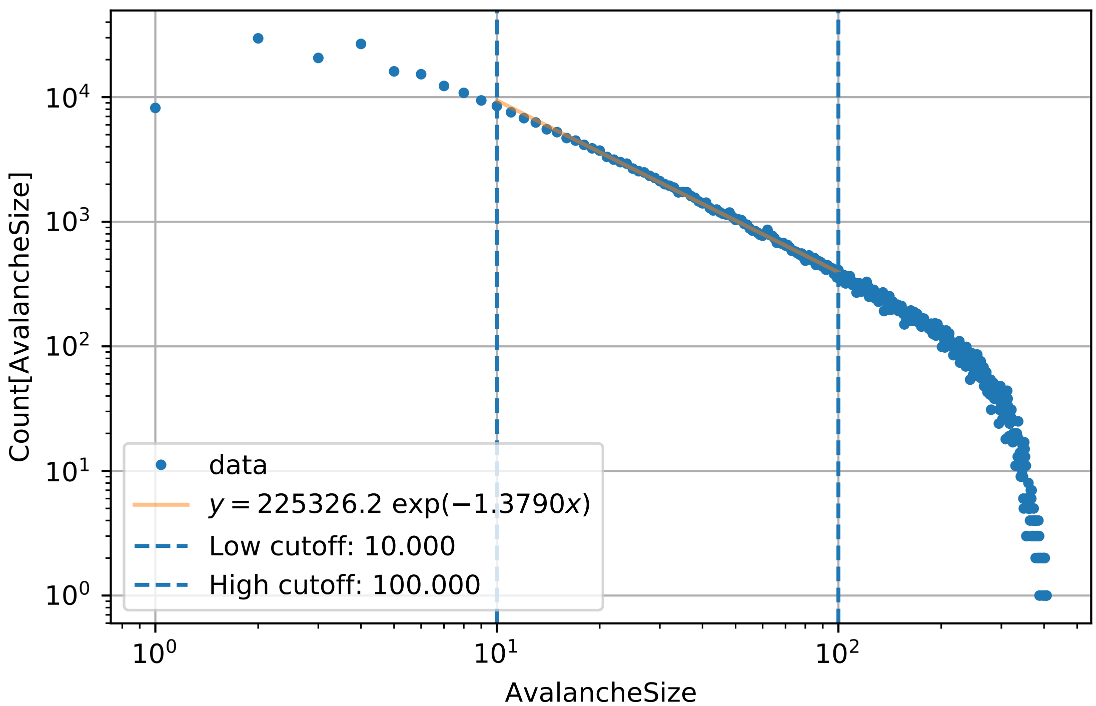

Figure 2: Number of the avalanches with the size given on the abscissa for the abelian Manna model with $L = 30$, $N = 5 \cdot 10^5$ and thermalization surplus $5 \cdot 10^4$

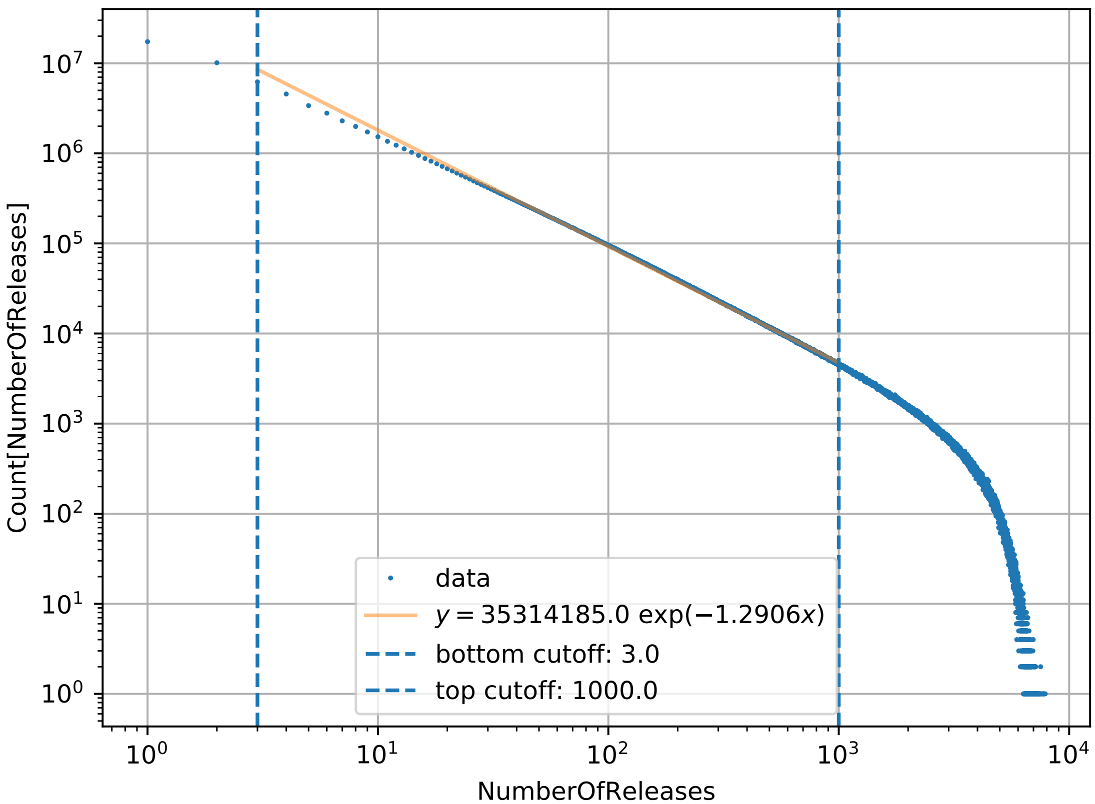

Figure 3: Number of the avalanches with number of releases given on the abscissa for the OFC model with $\alpha = 0.25$, $L = 35$, $N = 10^8$ and thermalization surplus 10^6

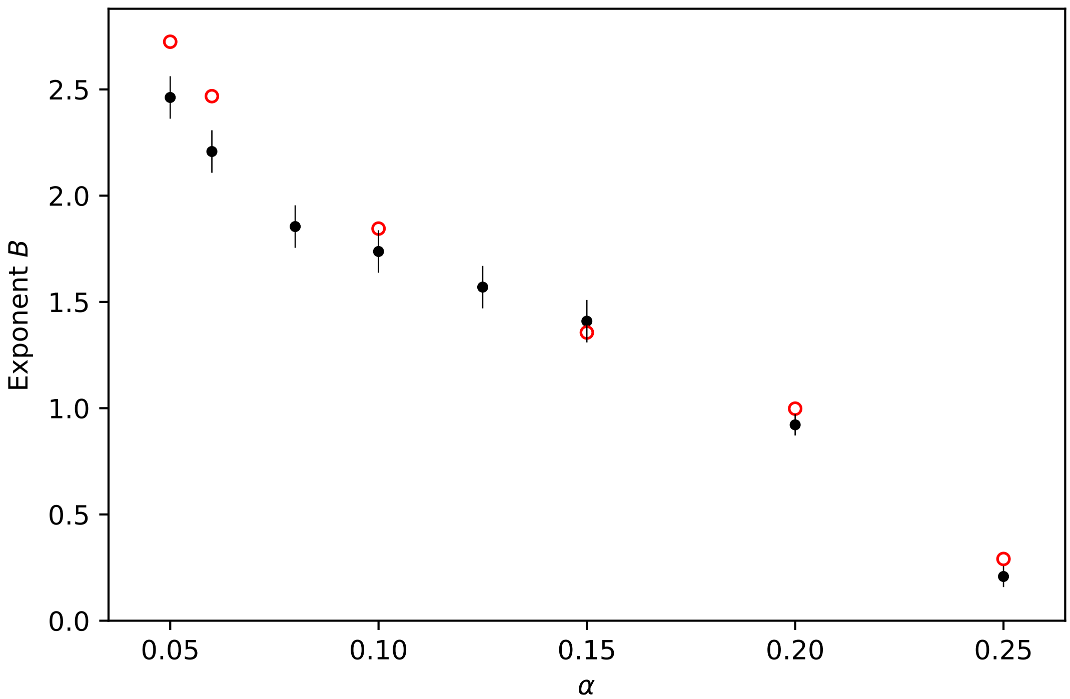

Figure 4: Comparison of results from the current code and results from original paper by Olami *et al.* α is the conservation level. The exponent B in [6] is such, that the probability density of having an earthquake of energy E, $P(E_0 = E)$, is proportional to $E^{-(1+B)}$. Black dots with error bars represent data extracted from Fig. 2 in [6] – the simulations made for 35×35 system with a series of values of α. Olami *et al.* take total number of relaxations as proportional to the energy released during a given earthquake. Red circles represent values of B calculated for the apparent exponents (ie. $B = \tau - 1$ if τ is an apparent exponent for number of releases, where each release (relaxation) is a transfer of energy from one active site to its neighbours) from exemplary simulations made by the SocSIM code for a 35×35 system. The simulations for $\alpha = \{0.25,\ 0.2,\ 0.15\}$ were performed for $N = 10^8$ and a simulation for $\alpha = 0.1$ was performed for $N = 2 \cdot 10^8$. The simulations with $\alpha = \{0.06,\ 0.05\}$ were performed for $N = 1.5 \cdot 10^8$; however higher values of N would be suggested for these highly dissipative cases. All six simulations had thermalization surplus $1\% \cdot N$. Plots for the simulations can be found here

References

[1] Per Bak, Maya Paczuski. Complexity, contingency, and criticality. *Proceedings of the National Academy of Sciences*, 92(15):6689–6696, 1995.

[2] Gunnar Pruessner. *Self-Organised Criticality: Theory, Models and Characterisation*. Cambridge University Press, 2012.

[3] Per Bak, Chao Tang, Kurt Wiesenfeld. Self-organized criticality: An explanation of the 1/f noise. *Phys. Rev. Lett.*, 59:381–384, Jul 1987.

[4] Subhrangshu Sekhar Manna. Two-state model of self-organized criticality. *Journal of Physics A: Mathematical and General*, 24(7):L363–L369, apr 1991.

[5] Deepak Dhar. The abelian sandpile and related models. *Physica A: Statistical Mechanics and its Applications*, 263(1-4):4–25, Feb 1999.

[6] Zeev Olami, Hans Jacob S. Feder, Kim Christensen. Self-organized criticality in a continuous, nonconservative cellular automaton modeling earthquakes. *Phys. Rev. Lett.*, 68:1244–1247, Feb 1992.

[7] Per Bak, Kan Chen, Chao Tang. A forest-fire model and some thoughts on turbulence. *Physics Letters A*, 147(5):297 – 300, 1990.

[8] S. van der Walt, S. C. Colbert, G. Varoquaux. The numpy array: A structure for efficient numerical computation. *Computing in Science Engineering*, 13(2):22–30, March 2011.

[9] Siu Kwan Lam, Antoine Pitrou, Stanley Seibert. Numba: A llvm-based python jit compiler. *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, LLVM '15, New York, NY, USA, 2015. Association for Computing Machinery.

[10] Wes McKinney. Data structures for statistical computing in python. Stéfan van der Walt, Jarrod Millman, redaktorzy, *Proceedings of the 9th Python in Science Conference*, strony 51 – 56, 2010.

[11] Dask Development Team. *Dask: Library for dynamic task scheduling*, 2016.

[12] Gunnar Pruessner. SOC computer simulations. https://arxiv.org/abs/1301.2918, 2013.