Complexity Project Notes

Prof. Kim Christensen
Centre for Complexity Science
Imperial College London

9th January 2019

General Comments

Assessment.
Projects: The two projects – **Complexity Project** and **Networks Project** – carry equal weight. The projects will contribute 90% of the final mark for the Complexity & Networks course and four on-line blackboard tests during the term will each contribute 2.5%. There are no further exams for the Complexity & Networks course.
Projects: The marks quoted in the projects should be taken as a rough guideline only. What we are looking for in the projects is a demonstration that students have understood the concepts and problems. The bulk of marks are for successful (correct) programming (i.e., not the style of programming), good theoretical derivations, sensible data handling, interpretations and explanations of results. This means that clever or sophisticated programming is only rewarded indirectly (i.e., an efficient programme will enable you to produce results faster or obtain better statistics, provided you wrote and tested the code in a reasonable time).
Feedback: We aim at giving you feedback on the Complexity Project Report by Wednesday 13 March (3 weeks rather than 2 weeks after the deadline due to the large number of reports that has to be assessed), that is, two weeks before the Networks Project Report is due.

Numerical Tools. The primary language is Python. Libraries and functions relevant to the projects will be provided for Python via Blackboard, Complexity & Networks (2018-2019) under 'Complexity - Kim Christensen', 'Complexity Project' or 'Networks - Tim Evans', 'Networks Project', respectively. Students may, in principle, use any computer language or package to work in, and to produce results on Physics Computer Lab PCs or on their own machines. Demonstrators will only offer support in languages and packages in which they have experience. You are likely to find demonstrators have expertise in some other combinations, e.g., C++ and MATLAB. (NB: For the Networks Project, languages other than Phyton are not supported nor recommended as special library files are needed. Other languages are allowed, however, Tim strongly recommends that students discuss non-Phyton choices with him.) You should search the web for helpful sites and build up a collection of helpful files but please make sure to fully reference such contributions.

Blackboard Submitting. You should submit your project reports and the code used to produce results for marking via Blackboard so make sure you have access to Blackboard. If you do not have access to Complexity & Networks (2018-2019) on Blackboard, please contact Kayleigh or Amy at the UG office via (phugadm@). Please note that the submitted report and code will be subject to an automatic check for plagiarism.

Deadlines: The deadline for submitting the Complexity Project Report + code is MONDAY 18 FEBRUARY at 5:00PM. The deadline for submitting the Networks Project Report + code is MONDAY 25 MARCH at 5:00PM. College policy is that any work submitted up to 24 hours after the deadline will be capped at the pass mark while any work submitted more than 24 hours after the deadline will incur zero marks unless you have a valid excuse. Please remember that problems associated with computers are not accepted as valid excuses.

Once the Complexity Project Report + code is submitted, you are committed to the course, that is, you may no longer opt out.

1 Complexity Project Notes

1.1 Self-Organised Critical Model

The aim is to study the Oslo model, which is one of the simplest models displaying self-organised criticality. The Oslo model was first published by Christensen *et al.* (1996) [1, 2, 3]. Despite its simplicity, the Oslo model is rich and non-trivial in its behaviour, and its avalanche-size probability is consistent with the general framework for scaling and data collapse, that is, the hallmarks of a system displaying self-organised criticality.

The objective is to reinforce and consolidate your learning on self-organisation and criticality, in particular finite-size scaling by practical applications of the theory.

Algorithm of the Oslo model

Consider a $d = 1$ lattice composed of L sites, $i = 1, 2, \ldots, L$. The number of grains at each site i is referred to as the height h_i, while the (local) slope at site i is defined by $z_i = h_i - h_{i+1}$ where $h_{L+1} = 0$. Because the heights are integers (number of grains), the slopes are also integers. Each site i is assigned a threshold slope $z_i^{\text{th}} \in \{1, 2\}$ at random. The system is driven by adding grains, one-by-one, to the boundary site $i = 1$. A site relaxes when $z_i > z_i^{\text{th}}$ by letting a grain topple from site i to site $i+1$, that is, $h_i \to h_i - 1, h_{i+1} \to h_{i+1} + 1$ and then randomly selecting its threshold slope $z_i^{\text{th}} \in \{1, 2\}$. The dynamics in terms of slope-units of the boundary-driven Oslo model is then defined via the following algorithm:

1. *Initialisation.* Prepare the system in the empty configuration with $z_i = 0$ for all i and chose random initial threshold slopes $z_i^{\text{th}} \in \{1, 2\}$ for all i.

2. *Drive.* Add a grain at the left-most site $i = 1$:
$$z_1 \to z_1 + 1. \tag{1}$$

3. *Relaxation.* If $z_i > z_i^{\text{th}}$, relax site i.
For $i = 1$:
$$\begin{aligned} z_1 &\to z_1 - 2, \\ z_2 &\to z_2 + 1. \end{aligned} \tag{2a}$$
For $i = 2, \ldots, L-1$:
$$\begin{aligned} z_i &\to z_i - 2, \\ z_{i\pm1} &\to z_{i\pm1} + 1. \end{aligned} \tag{2b}$$
For $i = L$:
$$\begin{aligned} z_L &\to z_L - 1, \\ z_{L-1} &\to z_{L-1} + 1. \end{aligned} \tag{2c}$$
Choose a new threshold slope $z_i^{\text{th}} \in \{1, 2\}$ at random **for the relaxed site only**, i.e., let
$$z_i^{\text{th}} = \begin{cases} 1 & \text{with probability } p \\ 2 & \text{with probability } 1-p \end{cases} \tag{2d}$$
using $p = 1/2$. Continue relaxing sites until $z_i \leq z_i^{\text{th}}$ for all i.

4. *Iteration.* Return to **2.**

We define the **avalanche size** s as the total number of relaxations initiated by adding one grain at the boundary $i = 1$. Note that <u>includes</u> avalanches of size $s = 0$.

1.1.1 Implementation of the Oslo model. [Marks: 5/100].

Write a computer programme (using the programme language of your choice) to implement the algorithm of the Oslo model in which you can easily change the system size L.

Typical mistakes in implementations:

(1) You should only reset the threshold slope z_i^{th} for the sites i that have toppled/relaxed.

(2) Ensure that the threshold slopes $z_i^{\text{th}} = 1$ and 2 are chosen with probability $p = 1/2$. NB: This does not imply that in a stable configuration these threshold appear with equal probability - why not?

(3) Ensure that you relax all sites in the system before adding a new grain.

(4) Ensure that you implement the relaxation algorithm correctly for bulk sites $i = 2, 3, ..., L-1$ and the two boundary sites $i = 1$ and $i = L$, respectively.

TASK 1 [5 marks]: Devise and perform some simple tests (e.g. by selecting particular simple values of p) to check whether your programme is working as intended. Document your tests in the project report.

IMPORTANT: Ideally, you should have implemented successfully the Oslo Model and performed TASK 1 by the start of the computing classes Wednesday 23 January 10am.

Comment: It is important that you devise and document your own tests to check if the implementation is correctly done, e.g. by choosing particular values of the probability p. For $p = 1/2$ (the original Oslo Model), you should also devise and document at least one test. Here, we list two measures that you can use to test your implementation for $p = 1/2$ (However, in addition, you **must** devise your own test!): Measure the height at site $i = 1$ averaged over time once the systems has reached the steady state. For $L = 16$ you should measure about 26.5 and for $L = 32$ about 53.9. If you do not measure these values, you can safely assume that something is wrong with your implementation.

1.1.2 The height of the pile $h(t; L)$ [Marks: 55/100].

We will first focus on measuring the height of the pile. Each of the measurements detailed below should be completed for the system sizes $L = 4, 8, 16, 32, 64, 128, 256$. If your programme is efficient, you may be able to simulate even larger system sizes $L = 512, \ldots$. It would make sense to implement ALL measurements and check (and double-check) that they make sense for a single system size before you begin the task of collecting measurements for all L.

Consider a system of size L and let the time t be measured in units of the number of grains added. Define the height at site $i = 1$ at time t as the height of the pile $h(t; L)$ when the pile has come to rest after adding the t'th grain. Because $z_i = h_i - h_{i+1}$ with $h_{L+1} = 0$, we find
$$h(t; L) = \sum_{i=1}^{L} z_i(t), \tag{3}$$
that is, the total height of the pile at time t is the sum of all the L local slopes $z_1(t)$, $z_2(t), \ldots, z_L(t)$ at time t in the pile.

Note that the total height of the system only changes when adding grains to the pile ($h_1 \to h_1 + 1$) or when site $i = 1$ topples ($h_1 \to h_1 - 1$). Therefore, when measuring the height, it might be more efficient to keep track of the changes of the height $h(t; L)$ during the avalanche rather than to use Eq.(3) to measure the height after each avalanche.

TASK 2a [5 marks]: Starting from an empty system, measure the total height of the pile as a function of time t for the range of system sizes listed above. Plot the height $h(t; L)$ vs. time t for the various system sizes in the same plot. Reflect upon the results obtained in terms of transient and recurrent configurations.

TASK 2b [5 marks]: In this task, we are only interested in the scaling behaviour for $L \gg 1$, that is, we may ignore potential corrections to scaling. Devise a **theoretical** argument to show that for a system of size L in the steady state, you expect the average of the height of the pile over time to scale linearly with system size. Similarly, argue theoretically that the average of the cross-over time $\langle t_c(L)\rangle$ for systems of linear size L to reach the steady state (set of recurrent configurations) scales like L^2.

For TASK 2c, you may want to smooth out the data by taking the average over M different realisations (i.e., use different random number sequences) for a given system size Define

$$\tilde{h}(t; L) = \frac{1}{M} \sum_{j=1}^{M} h^j(t; L), \tag{4}$$

where $h^j(t; L)$ is the height at time t in the jth realisation of a system of size L.

TASK 2c [10 marks]: Guided by your answers to the two questions in TASK 2b, produce a data collapse for the processed height $\tilde{h}(t; L)$ vs. time t for the various system sizes. Explain carefully how you produced a data collapse and express that mathematically, introducing a scaling function $\mathcal{F}$: $\tilde{h}(t; L) = \textit{something}\, \mathcal{F}\,(\textit{argument})$, identifying '*something*' and the '*argument*'. How does the scaling function $\mathcal{F}(x)$ behave for large arguments $x \gg 1$ and for small arguments $x \ll 1$ and why must it be so? From this result, obtain/predict how $\tilde{h}(t; L)$ increases as a function of t during the transient.

We will now investigate more carefully the scaling for the cross-over time $\langle t_c(L)\rangle$ averaged over systems of size L by attempting to include corrections to scaling. For a system of size L, define the cross-over time as the number of grains in the system before an added grain induces a grain to leave the system for the first time, that is, $t_c(L) = \sum_{i=1}^{L} z_i \cdot i$, where z_i are the local slopes in the system to which an added grain induces a flow out of the system for the first time.

TASK 2d [10 marks]: Assuming that $\langle z_i\rangle = \langle z\rangle$, show theoretically that $\langle t_c(L)\rangle = \frac{\langle z\rangle}{2} L^2 \left(1 + \frac{1}{L}\right)$. By numerically measuring $t_c(L)$ as the number of grains in the system before an added grain induces a grain to leave the system for the first time, starting from an empty system, estimate the average of the cross-over time as $\langle t_c(L)\rangle$. Demonstrate whether your data corroborate your theoretical prediction.

We will now investigate more carefully the scaling of the average height in a system of size L, attempting to reveal numerically corrections to scaling. Only consider the height $h(t; L)$ (not $\tilde{h}(t; L)$) for times $t \geq t_0$ where $t_0 > t_c(L)$, that is, the system has reached the attractor of the dynamics. For each system of size $L = 4, 8, 16, 32, 64, 128, 256$, measure

1. The average height (height averaged over time), that is,

$$\langle h(t; L)\rangle_t = \lim_{T\to\infty} \frac{1}{T} \sum_{t=t_0+1}^{t_0+T} h(t; L). \tag{5}$$

2. The standard deviation of the height, that is,

$$\begin{aligned} \sigma_h(L) &= \sqrt{\langle h^2(t; L)\rangle_t - \langle h(t; L)\rangle_t^2} \\ &= \sqrt{\lim_{T\to\infty} \frac{1}{T} \sum_{t=t_0+1}^{t_0+T} h^2(t; L) - \left[\lim_{T\to\infty} \frac{1}{T} \sum_{t=t_0+1}^{t_0+T} h(t; L)\right]^2}. \end{aligned} \tag{6}$$

3. The height probability

$$P(h; L) = \frac{\text{No. of observed configurations with height } h \text{ in pile of size } L}{\text{Total no. observed configurations}}, \tag{7a}$$

$$\sum_{h=0}^{\infty} P(h; L) = 1. \tag{7b}$$

TASK 2e [10 marks]: Now we consider the numerical data for the average height $\langle h(t; L)\rangle_t$ carefully to investigate whether it contain signs of corrections to scaling. Assume the following form of the corrections to scaling $\langle h(t; L)\rangle_t = a_0 L \left(1 - a_1 L^{-\omega_1} + a_2 L^{-\omega_2} + \ldots\right)$ where $\omega_i > 0$ and a_i are constants. Neglecting terms with $i > 1$, can you devise a procedure to estimate a_0 and ω_1 using your measured data?

TASK 2f [5 marks]: How does the standard deviation of the height $\sigma_h(L)$ scale with system size? From your answers to TASKS 2e and 2f, can you predict what will happen with the average slope and its standard deviation in the limit of $L \to \infty$?

TASK 2g [10 marks]: We consider $h = \sum_{i=1}^{L} z_i$. Assume that z_i are independent, identically distributed random variables with finite variance. Theoretically then, how would you expect $P(h; L)$ to be distributed when $L \gg 1$? Now plot the measured height probability $P(h; L)$ vs. h for various system sizes on the same plot. Inspired by the theoretical expectation for $P(h; L)$ under the assumptions above, use the *measured* values of $\langle h\rangle$ and σ_h to produce a data collapse for the various system sizes of $P(h; L)$. Explain carefully how you transformed (manipulated) the data to produce a data collapse and express that mathematically. Does the data collapse trace out the function expected from the theoretical prediction? What theoretical prediction does the assumption that z_i are independent, identically distributed random variables with finite variance yield for the scaling of $\sigma_h(L)$? Comment on your finding and the implications hereof.

1.1.3 The avalanche-size probability $P(s; L)$ [Marks: 40/100].

We will now focus on measuring the avalanche-size probability and associated moments. Consider a system of size L and define the avalanche size s as the number of relaxations (topplings) caused by addition of one grain at site $i = 1$. Note that includes zero avalanches $s = 0$ where no toppling is induced by adding a grain to the pile! These are important for the normalisation but are obviously omitted in the plotting if using a logarithmic scale.

Now only consider the avalanches after the system has reached the attractor of the dynamics, that is, only consider times $t > t_c(L)$.

Measure the normalised avalanche-size probability

$$P_N(s; L) = \frac{\text{No. of avalanches of size } s \text{ in a system of size } L}{\text{Total no. avalanches } N} \tag{8}$$

with $\sum_{s=0}^{\infty} P_N(s; L) = 1$. Read App. E in the lecture notes and apply the notion of data-binning to your avalanche-size probability (e.g. by uploading the file 'logbin6-2-2018.py' from 'Complexity Project' on Blackboard – please read usage instructions within the programme carefully).

TASK 3a [10 marks]: Using the log-binned data, plot the avalanche-size probabilities $\tilde{P}_N(s; L)$ vs. avalanche size s for all system sizes $L = 4, 8, 16, 32, 64, 128, 256$ on the same plot for a reasonably large N. State clearly how you are processing the data before plotting them. Describe qualitatively your results.

TASK 3b [10 marks]: Test if $\tilde{P}_N(s; L)$ are consistent with the finite-size scaling ansatz

$$\tilde{P}_N(s; L) \propto s^{-\tau_s} \mathcal{G}\left(s/L^D\right) \quad \text{for } L \gg 1, s \gg 1, \tag{9}$$

and estimate the values of the the avalanche dimension D and the avalanche-size exponent τ_s by performing a data collapse.

TASK 3c [20 marks]: Measure directly the k'th moment $\langle s^k \rangle$ for $k = 1, 2, 3, 4$ where

$$\langle s^k \rangle = \lim_{T \to \infty} \frac{1}{T} \sum_{t=t_0+1}^{t_0+T} s_t^k \tag{10}$$

where s_t is the measured avalanche size at time t and $t_0 > t_c(L)$, that is, the system has reached the steady state. Note that you should include avalanches of size zero $s_t = 0$.

Assuming the finite-size scaling ansatz given in Eq. (9), how does the k'th moment scale with system size L? Show that the data are consistent with your prediction of the scaling and use the moment scaling analysis method to estimate the values of the avalanche dimension D and the avalanche-size exponent τ_s. Explain concisely the procedure applied and present your results in relevant plots, reflect and comment on your findings. Demonstrate whether there are corrections to scaling in your numerical data for the moments.

References

[1] K. Christensen, A. Corral, V. Frette, J. Feder, and T. Jøssang, *Tracer dispersion in a self-organized critical system*, Phys. Rev. Lett. **77**, 107–110 (1996).

[2] K. Christensen and N.R. Moloney, *Complexity and Criticality*, ICP (2005).

[3] G. Pruessner, *Self-organised Criticality*, Cambridge University Press (2012).