Note on Toom's two-line voting rule

Ethan Lake

This note contains a numerical exploration of the "two-line voting" (TLV) 1d cellular automaton introduced by Toom, who was motivated to construct an eroder with more symmetries than the GKL automaton. Toom showed that this model is an eroder, and the only other work I am aware of on this model is Kihong Park's PhD thesis, wherein it was proven that the TLV automaton is ergodic under p-bounded noise. In this note we will understand why this is so, and verify ergodicity numerically.

The update rule and symmetries

The TLV automaton is essentially a more symmetric version of the GKL automaton. When formulated as a two-state automaton for spins $s_i \in \{0,1\}$ on a chain of size $2L$, the update rules are as follows:

$$s_i \to \begin{cases} \text{maj}(s_{i-1}, s_{i+2}, s_{i+4}) & i \in 2\mathbb{Z}+1 \\ \text{maj}(s_{i+1}, s_{i-2}, s_{i-4}) & i \in 2\mathbb{Z} \end{cases} \tag{1}$$

From this description it is not immediately clear what to expect. To rephrase things, we note that as this rule has a 2-site unit cell, it is hepful to instead write things as a 4-state automaton acting on a chain of length L. We will use the following notation:

$$\begin{aligned} |0\rangle_{2i} \otimes |0\rangle_{2i+1} &\to |0\rangle_i \\ |0\rangle_{2i} \otimes |1\rangle_{2i+1} &\to |A\rangle_i \\ |1\rangle_{2i} \otimes |0\rangle_{2i+1} &\to |B\rangle_i \\ |1\rangle_{2i} \otimes |1\rangle_{2i+1} &\to |1\rangle_i. \end{aligned} \tag{2}$$

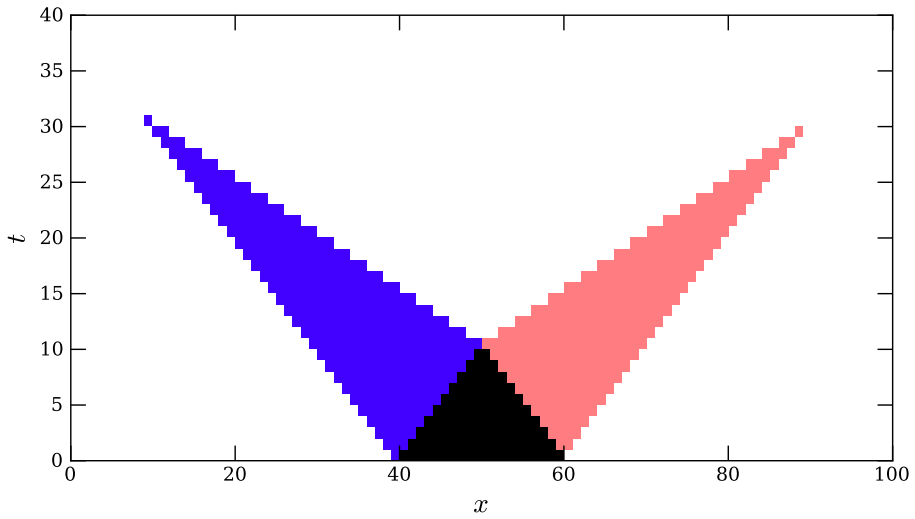
One checks that $|0\rangle^L, |1\rangle^L, |A\rangle^L$, and $|B\rangle^L$ are all fixed points of the noise-free update rule. The former two are stable fixed points, while the latter two are unstable. As we will see pictorially below, the A and B states are used to communicate messages between domain walls of 0 and 1, which facilitate error correction when errors occur on top of the two stable fixed points.

There are two symmetries of TLV, which we will call X (a global spin flip in the two-state notation), and R (a reflection combined with an exchange of the two logical states 0 and 1),

$$\begin{aligned} X &: A \leftrightarrow B,\ 0 \leftrightarrow 1 \\ R &: 0 \leftrightarrow 1,\ \text{reflection} \end{aligned} \tag{3}$$

Spacetime domain junctions

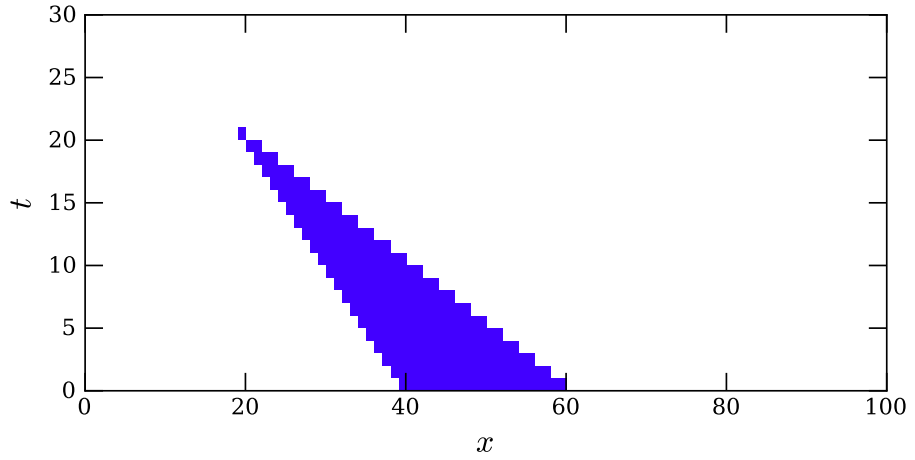
To understand how TLV works, we will graphically examine the allowed rules for patching together different domains in spacetime. Letting 0s be white, 1s black, As purple, and Bs pink, a single minority domain of 1s in a background of 0s is eroded as such:



(4)

Either the X or R symmetries show that the pattern for the erosion of a minority domain of 0s is identical after interchanging A and B. The slope-1 boundaries between the 0/1 and A/B regions should be viewed as messages that domain walls send out in order to identify partner domain walls to pair with. When these messages meet, they are converted into "cleaner" messages that move twice as fast (the slope-1/2 boundaries), which erase the initial messages that traveled in the wrong direction.

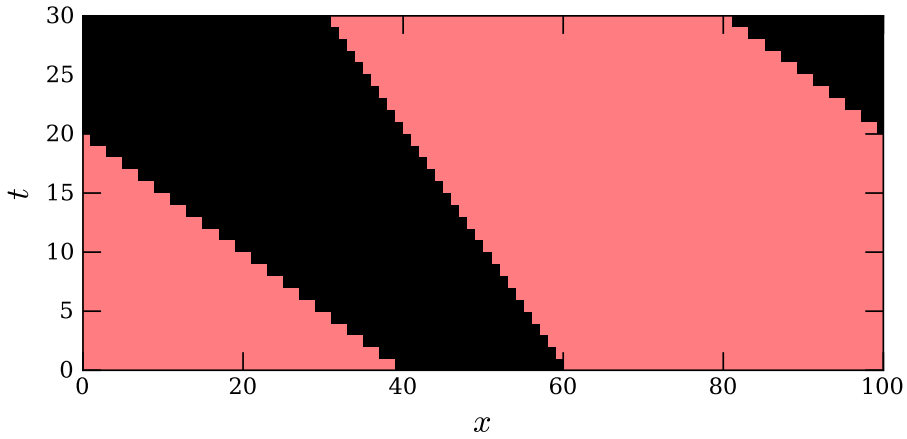
To get used to the dynamics, some more examples follow. A single domain of As is eroded from one side in a GKL-esque way as



(5)

A minority domain of 1s in a background of As infects them as follows (which can be

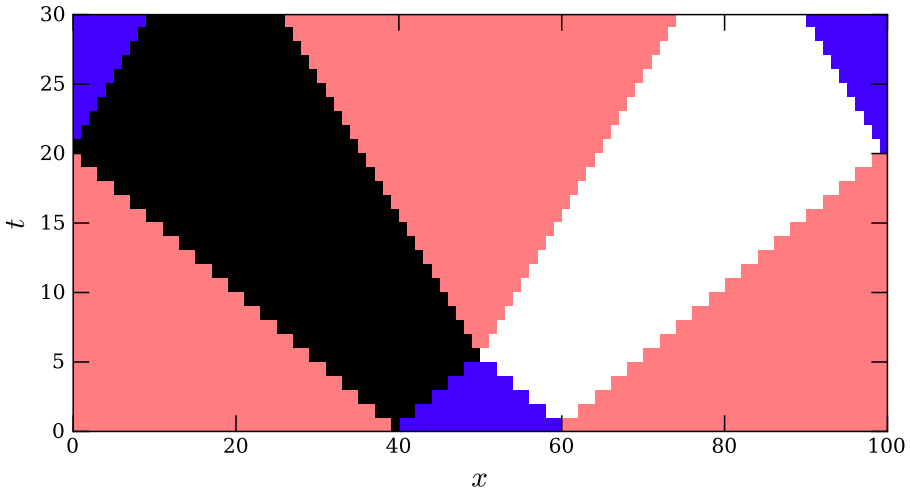
inferred from the previous picture):



(6)

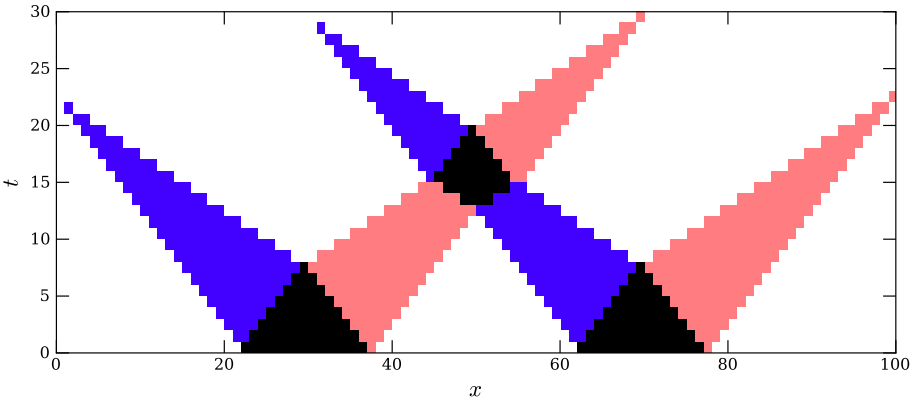
where we are operating with PBC.

Finally, a minority domain of Bs in a background of As creates growing regions of 1s and 0s:



(7)

These rules (and their symmetry-related variants) can be used to understand more complicated scenarios. For example, the pattern created when two A and B regions intersect during the course of error correction is as follows:



(8)

Failure as a memory

Let us now understand why TLV is an eroder, but ergodic in the presence of transient noise. First let us recall a sufficient condition for non-ergodicity. Very schematically, let us group a given spacetime noise realization into errors of different sizes (how to do this rigorously is explained in a different note). An automaton will be non-ergodic if:

- isolated errors at scale l are cleaned up in a time $\sim l^z$ for some (probably small-ish) constant z, and
- an error at scale l which occurs inside a spacetime region in which an error at a larger scale l' is being cleaned up slows down the correction of the large-scale error by at most an l'-independent time scaling as $\lesssim l^z$.

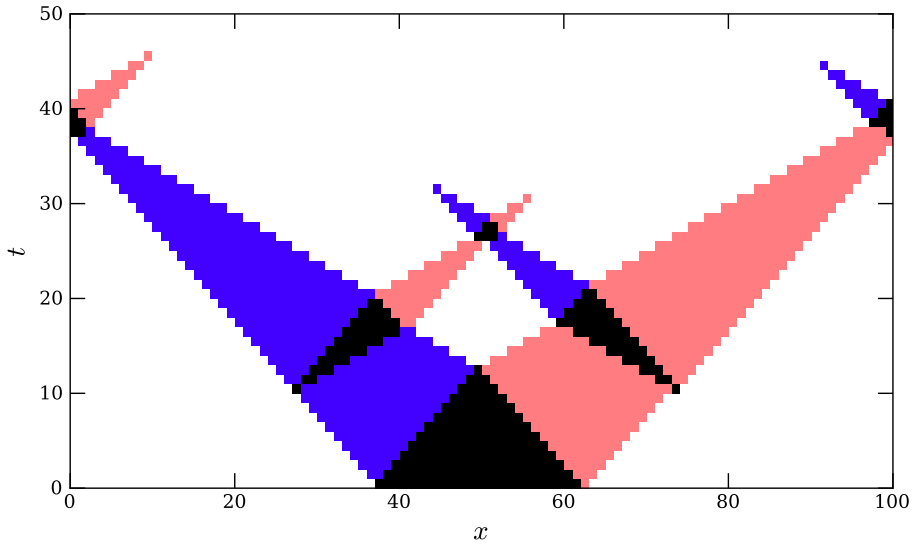
For an asynchronous automaton, these bullet points should be interpreted as holding with high probability. I currently think that these conditions are also necessary for ergodicity breaking. Note that for the second bullet point, we do not require that the spacetime influence of the l-scale error be independent of l'; indeed generically the size of the l-scale error's influence will increase with l'. Rather, we only require that the slowdown in the l' error's cleanup time be independent of l'.[1]

Ergodicity in the TLV automaton is ultimately a consequence of the instability of the message regions (A and B regions) with respect tological region inclusions (0 and 1 regions). This instability allows small-scale errors to boostrap off of large-scale errors, violating the second bullet point above. For TLV $z = 1$, and a local transient error at scale l which occurs during the correction of an error at scale l' can delay the correction of a large error by a time of order $\sim l'$, even in the limit $l/l' \to 0$.

As an example, consider again the erosion of a single large minority domain of 1s in a 0 background. Flipping single 0s to 1s near the boundary of the spacetime support of the A and B messages can create a time delay in the A/B message erosion of a

[1] As an example, consider Toom's rule, with a large error over a square of size l', and a small error which enlargens the bottom-left corner of this square by an amount $\sim l$. The small error will remain present until the big error is cleaned up—so that its temporal support scales like l'—but the large error's cleanup time will only be delayed by $\sim l$, independent of l'.

magnitude proportional to its erosion time in the absense of errors:

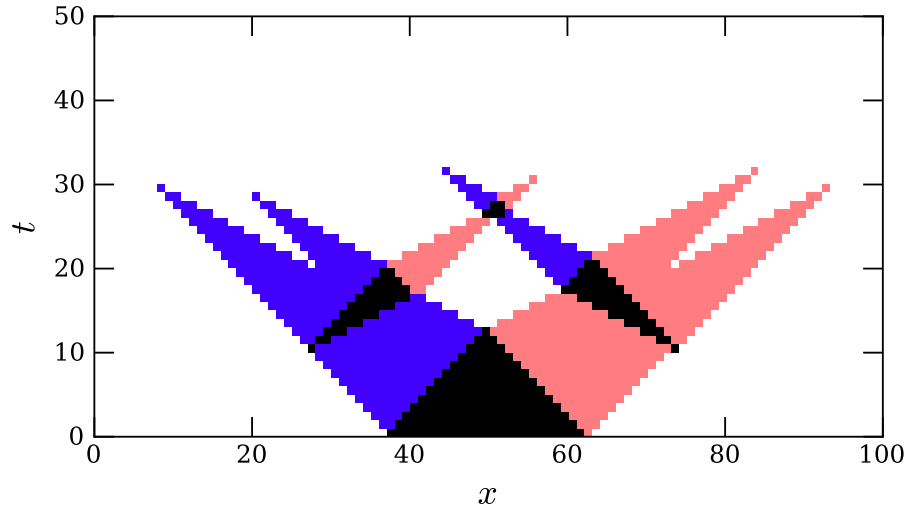


(9)

where the errors occur at time $t = 10$. We see that the erosion of the large A/B message regions is delayed by a time proportional to their 'thickness'; hence the statement above.[2]

As a concrete illustration of how these delays affect error cleanup, consider an error model where each site is subjected to i.i.d random errors (i.e. with probability p, the spin at a given site is randomized to one of $0, 1, A, B$). We obtain things like the

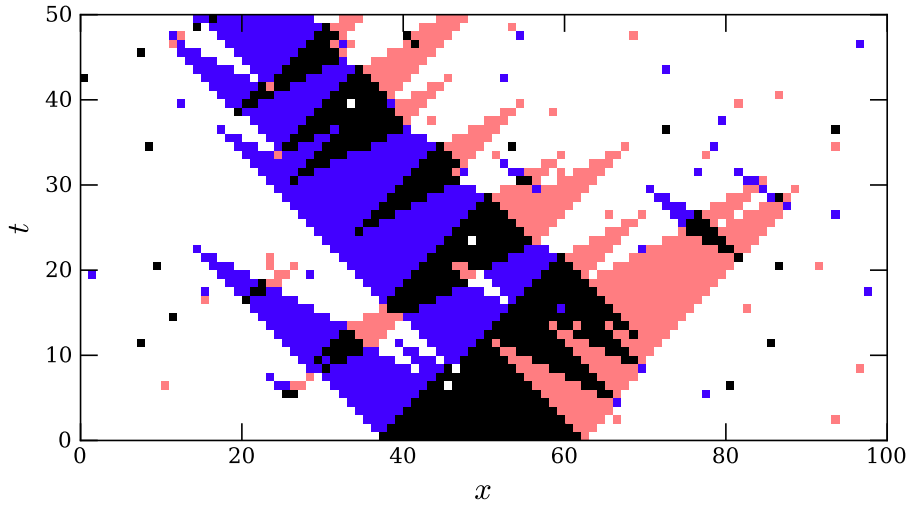
[2] A potential counter-argument would be that errors in which A or B flip to 0 will *hasten* message erosion by a similarly large factor (in the above example where 0 is the majority), e.g.



(10)

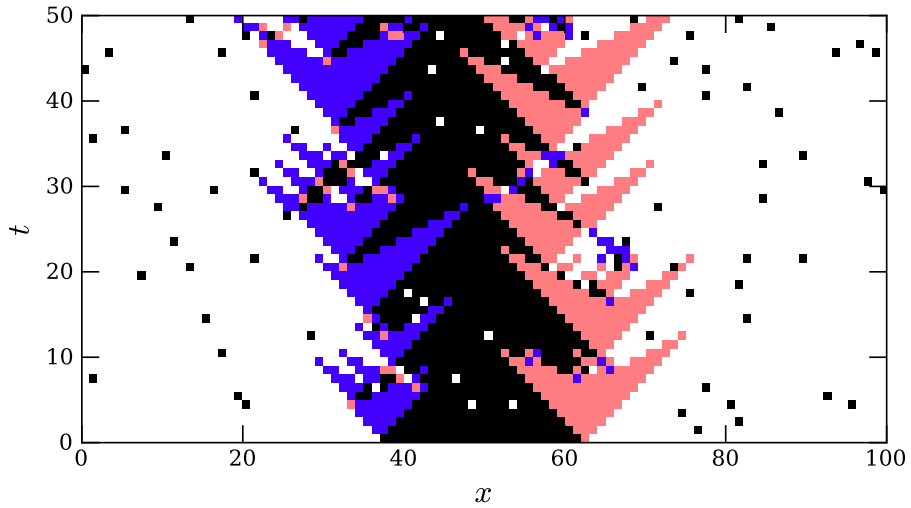
However, even if we assume that the type of errors are not chosen adversarially, having random processes which increase / decrease cleanup times of large errors in this way is probably not good.

following: (here $p = 0.025$)



(11)

The problem does not go away under any reasonable (viz, interesting) modification of the noise model. For example, one may assume that the relative parity between bits in the same unit cell (in the two-state notation) is not subject to errors; we will call this the "restricted" noise model. This allows the noise to act only as $A \leftrightarrow B, 0 \leftrightarrow 1$, and is an attempt to do something along the lines of "make the message subspace reliable, and let errors only act on the subspace that encodes the information". However, from the above pictures, it should be clear that this does not help. For noise strength $p = 0.05$, an example is:



(12)

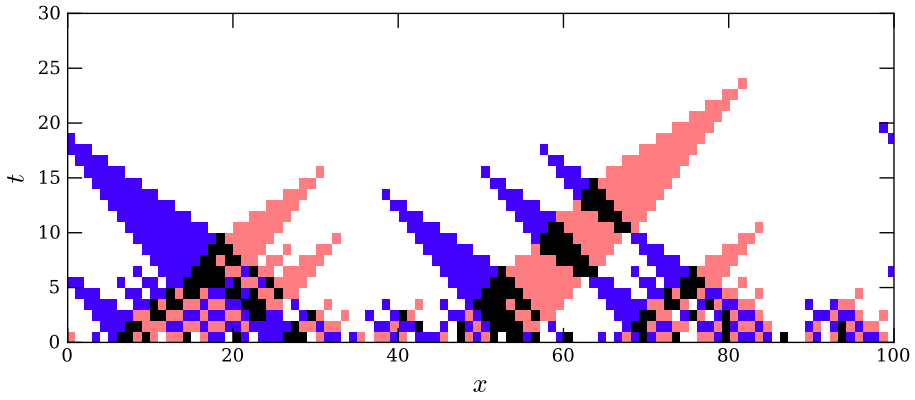
Erosion

Erosion

Despite the above problems, it should by now be reasonable that TLV is an eroder for both logical states 0^L and 1^L. While smaller errors in the initial state can still intersect with the message regions produced by larger errors in the initial state, these smaller

errors must be very close to the larger error in order to do anything; the further away from the larger error they are, the bigger these smaller errors must be. ethan: *this is not explained well but I think it can be made rigorous if really desired*

Numerics reveal it to in fact be a remarkably good eroder; as an example, below we show a randomly chosen state with magnetization density $+1/25$, which is correctly (and rather rapidly) eroded:



(13)

Relaxation Times

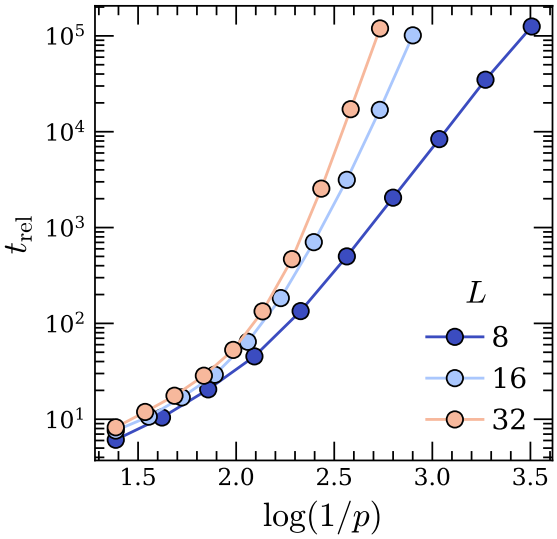
We can put this intuition to the test by numerically evaluating the relaxation time of the memory using Monte Carlo. We will use i.d.d bit flip errors with strength p, with t_{rel} defined as the expected time for a logical failure to occur. Since t_{rel} is calculated by running the automaton and asking when the ideal decoder fails, it will depend on L even in the trivial case where no error correction is performed. In this trivial case, one can argue that (see other notes)

$$t_{\text{rel}} \sim \frac{1}{p} \log(L). \tag{14}$$

Super-logarithmic growth of t_{rel} is thus our requirement for something to constitute a memory.

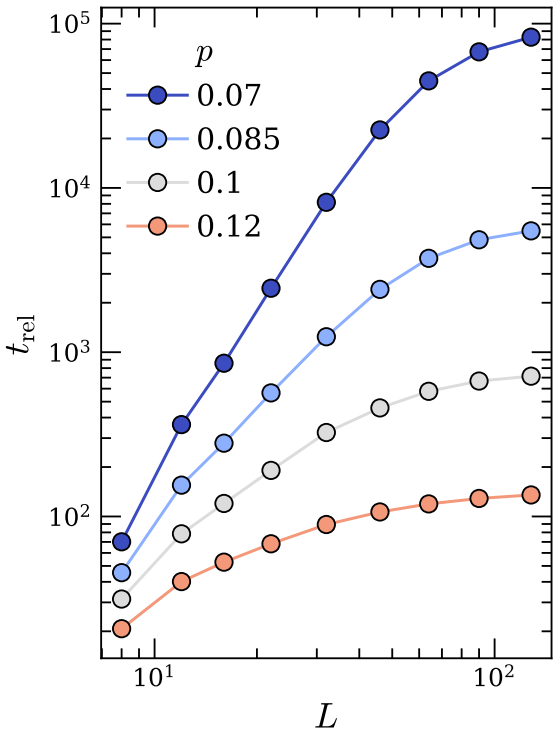
unrestricted noise

Consider first unrestricted noise. t_{rel} is numerically very large, with a perhaps promising-ish looking scaling against p at small L:



(15)

When we look at the scaling of t_{rel} against L however, we see that in the range $p \lesssim \exp(-2.5) \approx 0.08$ where we might suspect a threshold, the scaling flattens off after large enough L:

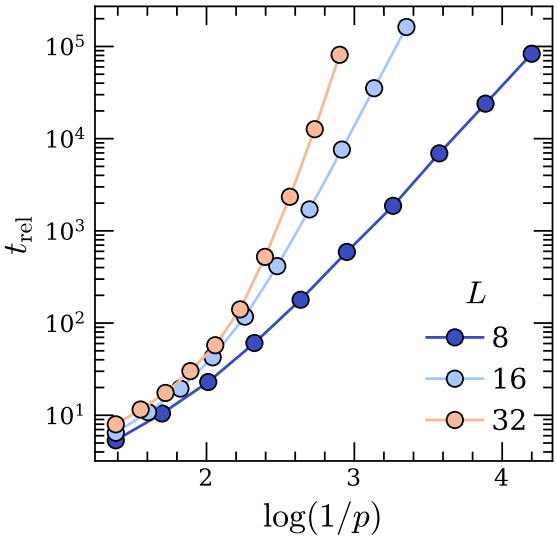


(16)

in keeping with the model's ergodicity.

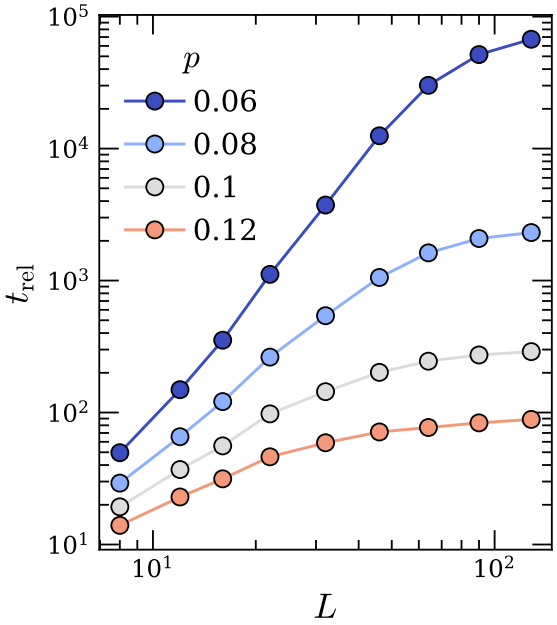
restricted noise

For completeness we can also try restricted noise, which acts only as $A \leftrightarrow B, 0 \leftrightarrow 1$. We said above that this shouldn't help; let us now verify this. t_{rel} against p again looks promising at small L:



(17)

When plotted against L though, the situation is quite similar to the case with unrestricted noise:



(18)