

# Supplementary Material: Numerical study of quantum self-correction in the 3D Cubic Code model

Sergey Bravyi<sup>1</sup> and Jeongwan Haah<sup>2</sup>

<sup>1</sup>*IBM Watson Research Center, Yorktown Heights, NY 10598*

<sup>2</sup>*Institute for Quantum Information and Matter,  
California Institute of Technology, Pasadena, CA 91125*

(Dated: 7 August 2013)

Here, we explain details of our numerical simulation. The interaction of the memory system with a thermal bath is simulated by Metropolis evolution. As we wish to observe low temperature behavior we adopt continuous time algorithm by Bortz, Kalos, and Lebowitz (BKL) [1]. A pseudo-random number generation package `RngStream` by L'Ecuyer [2] was used. As before, the coupling constant in the Hamiltonian is set to  $J = 1/2$  so a single defect has energy 1. Although the 3D Cubic Code is inherently quantum, it is relevant to consider only  $X$ -type errors (bit flip) in the simulation, thanks to the duality of the  $X$ - and  $Z$ -type stabilizer generators of the 3D Cubic Code. The simulation thus is purely classical. The errors are represented by a binary array of length  $2L^3$ , and the corresponding syndrome by a binary array of length  $L^3$ .

The memory time is measured to be the first time when the memory becomes unreliable. There are two cases the memory is unreliable: either the decoding algorithm fails to remove all the defects so we have to reinitialize the memory, or a nontrivial logical error is occurred. It is thus necessary in our simulation to keep track of the error operator during the time evolution. In fact, most of the time, it was the decoding algorithm's failure that made the memory unreliable. Nontrivial logical errors occurred only for very small system sizes  $L = 5, 7$ .

It is too costly to decode the system every time it is updated. Alternatively, we have performed a trial decoding every fixed time interval

$$T_{ec} = \frac{e^{4\beta}}{100}$$

where  $\beta$  is the inverse temperature. Although the time evolution of the BKL algorithm is stochastic, a single BKL update typically advances time much smaller than  $T_{ec}$ . So it makes sense to decode the system every  $T_{ec}$ . The exponential factor appears naturally because BKL algorithm advances time exponentially faster as  $\beta$  increases. It is to be emphasized that we do not alter the system by the trial decodings (a copy of the actual syndrome has been created for each trial decoding).

The system sizes  $L^3$  for the simulation are chosen such that the code space dimension is exactly 2, for which the complete list of logical operators is known. If the linear size  $L$  is  $\leq 200$ , this is the case when  $L$  is not a multiple of 2, 15, or 63 [3]. For these system sizes, to check whether a logical operator is nontrivial is to compute the commutation relation with the known nontrivial logical operators.

The measured memory time for a given  $L$  and  $\beta$  is observed to follow an exponential distribution; a memory system is corrupted with a certain probability given time interval. Specifically, the probability that the measured memory time is  $t$  is proportional to  $e^{-t/\tau}$ . Thus the memory time should be presented as the characteristic time of the exponential distribution;  $T_{mem} = \tau$ . We choose the estimator for the characteristic time to be the sample average  $\bar{T} = \frac{1}{n} \sum_i^n T_i$ . The deviation of the estimator will follow a normal distribution for large number  $n$  of samples. We calculated the confidence interval to be the standard deviation of the samples divided by  $\sqrt{n}$ . For each  $L$ , 400 samples when  $\beta \leq 5.0$  and 100 samples when  $\beta > 5.0$  were simulated. The numerical data were obtained for 26 values of  $\beta$  in the interval  $4.0 \leq \beta \leq 5.25$  for each lattice size  $5 \leq L \leq 33$  satisfying  $k(L) = 2$ . That is,  $L$  is odd and  $L \neq 15$ . The result is summarized in Fig. 2, 3 in the main text.

Figure 3 clearly supports  $\log T_{mem} = \frac{1}{4}c\beta^2 + \dots$ . Figure 2 demonstrates the power law for small system size:

$$T_{mem} \propto L^{2.93\beta - 10.5}$$

We wish to relate some details of the model with the numerical coefficients. The Rigorous analysis of the previous section, gives a relatively small coefficient  $c$  of the energy barrier for correctable errors by our RG decoder. However, we expect that the coefficient of  $\beta$  in the exponent is the same as the constant  $c$  that appear in the energy barrier

$$E = c \log_2 R$$

to create an isolated defect separated from the other by a distance  $R$ . This is based on an intuition that the output  $P'$  of the decoder would have roughly the same support as the real error  $P$  for the most of the time, provided that the error has energy barrier less than  $\Delta = c \log_2 L_{tqo}$ . Thus, an error of energy barrier less than  $\Delta$  would be corrected by the decoder. Our empirical formula supports this intuition. It suggests that  $c = 2.93 \log 2 = 2.03 \sim 2$ .

Indeed, we can illustrate explicitly an error path that separates a single defect from the rest by distance  $2^p$  during which only  $2p + 4$  defects are needed. This is an improvement in the upper bound on the energy barrier for separating a charged set of defects, for which it was  $c \leq 4$  in [4].

Consider an error of weight 2 that creates 4 defects as shown in the top of Fig. 1. We call it the *level-0 hook*.

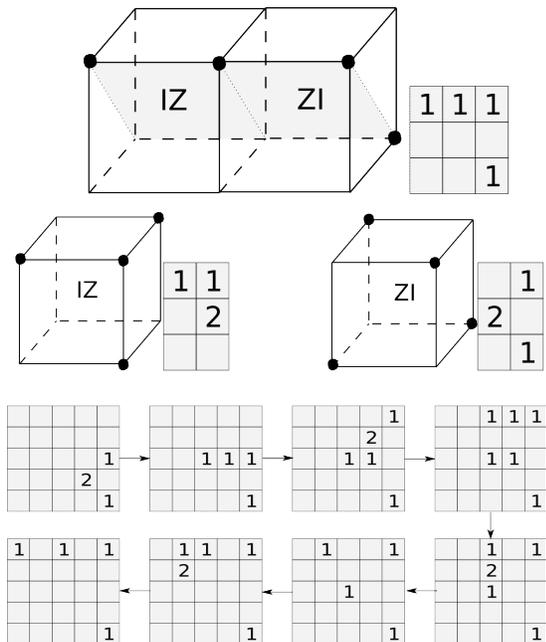


FIG. 1. Construction of a hook of level 2 from the vacuum. The grid diagram represents the position and the number of defects in the  $(x = z)$ -plane. For each transition, an operator of weight 1 is applied. The total number of defects never exceeds 6. From a level-0 hook (the second diagram in the sequence), a level-1 hook (the last in the sequence) is constructed using extra 2 defects.

The bottom sequence depicts a process to create a configuration shown at the bottom-left, which we call *level-1 hook*. One sees that level-1 hook is similar with ratio 2 to level-0, and is obtained from level-0 with extra 2 defects. One defines level- $p$  hooks hierarchically. We claim that a level- $p$  hook can be constructed from the vacuum using  $2p + 4$  defects. The proof is by induction. The case  $p = 1$  is treated in the diagrams. Suppose we can construct level- $p$  hook using  $2p + 4$  defects. Consider the  $2^{nd}$ ,  $4^{th}$ ,  $6^{th}$ , and  $8^{th}$  steps in Fig. 1. They can be viewed as a minuscule version of level- $p$  steps that construct a level- $(p + 1)$  hook from the level- $p$  hooks. It requires at most  $2p + 4 + 2$  defects to perform the level- $p$  step; this completes the induction. It may not be obvious whether a high level hook corresponds to a nontrivial logical operator, but such a large hook is bad enough to make our decoder fail.

- 
- [1] A. B. Bortz, M. H. Kalos, and J. L. Lebowitz, J. Comp. Phys. **17**, 10 (1975), ISSN 0021-9991.  
 [2] P. L'Ecuyer, R. Simard, E. J. Chen, and W. D. Kelton, Operations Research **50**, 1073 (2002), ISSN 0030364X,

- URL <http://www.jstor.org/stable/3088626>.  
 [3] J. Haah, Phys. Rev. A **83**, 042330 (2011).  
 [4] S. Bravyi and J. Haah, Phys. Rev. Lett. **107**, 150504 (2011).