Non-equilibrium criticality and efficient exploration of glassy landscapes with memory dynamics

Yan Ru Pei1,* and Massimiliano Di Ventra1,†

1Department of Physics, University of California, San Diego
La Jolla, CA 92093

Spin glasses are notoriously difficult to study both analytically and numerically due to the presence of frustration and metastability. Their highly non-convex landscapes require collective updates to explore efficiently. Currently, most state-of-the-art algorithms rely on stochastic spin clusters to perform non-local updates, but such “cluster algorithms” lack general efficiency. Here, we introduce a non-equilibrium approach for simulating spin glasses based on classical dynamics with memory. By simulating various classes of 3d spin glasses (Edwards-Anderson, partially-frustrated, and fully-frustrated models), we find that memory dynamically promotes critical spin clusters during time evolution, in a self-organizing manner. This facilitates an efficient exploration of the low-temperature phases of spin glasses.

I. INTRODUCTION

The study of spin glasses has contributed substantially to our understanding of a wide variety of phenomena1–3, much beyond the complex magnetic models for which they were first introduced4. In their most basic form, these systems are described by the following simple Hamiltonian5:

$$H = -\sum_{ij} J_{ij} s_i s_j,$$  \hspace{1cm} (1)

where the spins, arranged on some \(d\)-dimensional lattice, acquire the values, \(s_i = \pm 1\), and interact via coupling constants, \(J_{ij} = \pm 1\), with \(J\) a multivariate random variable taken from some distribution.

Despite the deceptively simple form, the energy landscape of the model Hamiltonian (1) is highly non-trivial6,7 for most conceivable distributions of \(J\). Decades of mathematical ingenuity have culminated in efficient (namely, polynomial-time) algorithms for computing the partition function of any realization of \(J\) in two dimensions8–11, but an efficient algorithm to simulate glasses in \(d > 2\) remains elusive. In fact, finding the ground state of a three-dimensional glass with arbitrary bonds was shown to be NP-complete12, with the task of computing its partition function shown to be NP-hard13. This hardness fundamentally limits the efficiency of any stochastic algorithm.

Earlier approaches for simulating the model Hamiltonian (1) were based on sequential Metropolis updates14,15 and modern extensions of this methodology have also been proposed16–20. However, these algorithms are generally plagued by a large dynamical critical exponent, making the simulation largely inefficient21.

Later on, a method based on the synchronous update of a large correlated cluster of spins was suggested22–24, which proved to be very effective for the 2d ferromagnetic Ising model. Unfortunately, despite several modifications made to account for the non-locality of frustration25,26, these cluster algorithms still struggle for high-dimensional glasses. The main reason behind their inefficiency is the tendency for the cluster percolation process to be persistently hyper-critical, due to a mismatch of critical temperature and cluster percolation ratio27,28. This means that the largest cluster component generally covers the entire lattice29, resulting in a trivial global spin-flip in most cases.

At the present stage, the best known method for tackling the above issues is to generate clusters based on replica overlaps (referred to as the “isenergetic cluster move” (ICM) method)30,31, which reverses the percolation ratio28. This is generally coupled with replica exchange methods such as parallel tempering (PT) for efficient thermalization32. Unfortunately, this approach relies heavily on the dimension of the lattice, and still tends to over-percolate in certain temperature ranges. Another recent trend is to use machine learning techniques to help identify efficient clusters33, by putting the hidden nodes on the edges (or plaquettes) of the lattice34. In some cases, the efforts towards this direction have been half-hearted attempts in under-employing the representative power35 of Boltzmann machines in modeling the Boltzmann distribution of the Ising glass, resulting in mathematically equivalent formulations of the traditional cluster algorithms25.

Here, instead, we propose a novel non-stochastic approach to efficiently learn the critical clusters of the glass during dynamics, without any algorithmic aid1. In sharp contrast to previous stochastic methods, which treat the spins and time as discrete variables, we first linearly relax1 the spin variables, and then couple them to mem-

*Email: yrpei@ucsd.edu
†Email: diventra@physics.ucsd.edu
II. MEMORY DYNAMICS

To introduce the memory dynamics, we first introduce the continuously relaxed spin glass Hamiltonian

$$H = - \sum_{ij} \left( J_{ij} \sigma_i \sigma_j - \frac{1}{2} \mu_{ij} (\sigma_i^2 + \sigma_j^2) \right), \quad \sigma_i \in [-1, +1],$$

where $\mu$ are non-uniform memory variables acting as dynamic Lagrange multipliers for constraints on the spin magnitude. If $\mu$ were fixed in time, then the standard dynamics

$$\partial_t \sigma_i = -\nabla_{\sigma_i} H = \sum_j \left( J_{ij} \sigma_j - \mu_{ij} \sigma_i \right),$$

would suffer from long auto-correlation times (critical slowing down), due to the presence of metastable states, even when the spins are continuously relaxed.

Therefore, in order to efficiently escape these states, we can continuously deform the energy local minima, and transform them into saddle points by letting the memory variables to evolve as

$$\partial_t \mu_{ij} = (J_{ij} \sigma_i \sigma_j - \gamma),$$

where $\gamma$ is some constant restricting the growth of $\mu_{ij}$, and $\mu_{ij} \in [0, 1]$. By providing dynamics to the variables $\mu_{ij}$, we see that the “gradient term” $\sum_j J_{ij} \sigma_j$ for the spins in Eq. (3) is compensated by the “cluster-like” update term $-\sum_j \mu_{ij} \sigma_i$ in the same equation (see discussion below).

By simulating the coupled Eqs. (3) and (4) until the system reaches a fixed time-out, we can take $s_i = \text{sgn}(\sigma_i)$ for a recorded state that minimizes the Ising energy in Eq. (1). Many numerical strategies can be used to improve the stability and convergence properties of the simulation. They are also included in the codes of the repository PeaBrane/Ising-Simulation, which can be used to directly reproduce Figs. 2 and 3. The particular numerical implementation we used in this work is given as

$$\dot{\sigma}_i = \alpha \sum_j J_{ij} \sigma_j - 2\beta \sum_j x_{ij} \sigma_i,$$

$$\dot{x}_{ij} = \gamma C_{ij} - y_{ij},$$

$$\dot{y}_{ij} = \delta x_{ij} - \zeta,$$

where $C_{ij} = \frac{1}{2} (J_{ij} \sigma_i \sigma_j + 1) \in [0, 1]$, $y$ is a secondary long-term memory ensuring stability, and $\alpha, \beta, \gamma, \delta, \zeta$ are time-scale parameters, fixed for all system sizes. We used the Euler method to integrate forward the above equations. More implementation details and the choice of parameters are given in the Supplementary Material (SM) Section D.
first assume that the memory variables are already at
as drawn in Fig. 1. process of forming open bonds in a
abilities proba-
dynamics to play a major role in the stability and efficiency of our
to tame numerical truncation errors
do not have to worry about using acceptance schemes

Before showing numerical results, we provide an understand-
ing of why such a memory dynamics would be effi-
cient in simulating spin glasses. First of all, we note that
the memory variables should not be considered as stan-
dard dual variables. Instead of being coupled to the
spin constraints (the second term of Eq. (2)), the memory
evolution is explicitly coupled to the state of interaction
between spins, $J_{ij}\sigma_i\sigma_j$, as written in Eq. (4), and this is
crucial for simulating frustrated systems. Furthermore,
since we are simulating the system at non-equilibrium, we
do not have to worry about using acceptance schemes

If we bound the memory variables between 0 and 1 (see
Section D in the SM), we can interpret them as proba-

To see why the memory dynamics are critical, let us
first assume that the memory variables are already at

The memory variables may also be defined on different unit
cells, such as on plaquettes for the fully frustrated Ising
model.

Note that, it is not necessary for us to algorithmically connect
the memory clusters and flip them discretely (see SM A), because
such cluster-update features are implicitly present in the equa-

FIG. 2: Simulations performed on 1000 fully-frustrated 3d Ising glasses (see SM F 2) sized $6^3$ to $12^3$. (Left) The effective
temperature, $T_{eff}$, of the memory dynamics is tracked in time (see SM F 3 for how we estimate this temperature) for the $6^3$
lattice. The critical temperature, $T_c$, of the fully frustrated glass is determined using the crossing of Binder’s cumulant (see SM
F 4). Note that the time it takes $T_{eff}$ to dive below $T_c$ is extremely short, less than 4 units of time, after which the memory
dynamics remain persistently below $T_c$. (Right) An arbitrary point in time $t_0 = 10^2$ is chosen, and the cluster size distribution
(CSD) is collected over the disorder realizations, and a $\Delta t = 2^8$ time window. Other than the tailing drop-off resulting from
finite-size effects, the CSD follows a power-law decay with the Fischer exponent being $\tau = 2.20 \pm 0.01$ for all simulated sizes.

See Fig. 5 in the SM for further empirical evidence that the Fischer exponent is insensitive to the underlying temperature and
lattice size, though it is dependent on the frustration profile of the underlying glass.

III. DYNAMICAL CRITICAL CLUSTERS

To provide additional evidence of criticality, we have numerically extracted the cluster size distribution (CSD)
for a fully-frustrated Ising model, as generated by the memory variables averaged over disorder at an arbitrary
point in time (see Fig. 2). While most state-of-the-art
algorithms fail to generate critical clusters even at the
critical temperature $T_c$, the memory clusters are
persistently critical at all temperatures (with the esti-
mation of non-equilibrium temperature outlined in SM
F 3). In other words, we see that the criticality is self-
organizing (SOC) through the entire simulation, and
it avoids critical slowing down at all points in time.

To provide additional evidence of criticality, we have numerically extracted the cluster size distribution (CSD)
for a fully-frustrated Ising model, as generated by the memory variables averaged over disorder at an arbitrary
point in time (see Fig. 2). While most state-of-the-art
algorithms fail to generate critical clusters even at the
critical temperature $T_c$, the memory clusters are
persistently critical at all temperatures (with the esti-
mation of non-equilibrium temperature outlined in SM
F 3). In other words, we see that the criticality is self-
organizing (SOC) through the entire simulation, and
it avoids critical slowing down at all points in time.

To provide additional evidence of criticality, we have numerically extracted the cluster size distribution (CSD)
for a fully-frustrated Ising model, as generated by the memory variables averaged over disorder at an arbitrary
point in time (see Fig. 2). While most state-of-the-art
algorithms fail to generate critical clusters even at the
critical temperature $T_c$, the memory clusters are
persistently critical at all temperatures (with the esti-
mation of non-equilibrium temperature outlined in SM
F 3). In other words, we see that the criticality is self-
organizing (SOC) through the entire simulation, and
it avoids critical slowing down at all points in time.

To provide additional evidence of criticality, we have numerically extracted the cluster size distribution (CSD)
for a fully-frustrated Ising model, as generated by the memory variables averaged over disorder at an arbitrary
point in time (see Fig. 2). While most state-of-the-art
algorithms fail to generate critical clusters even at the
critical temperature $T_c$, the memory clusters are
persistently critical at all temperatures (with the esti-
mation of non-equilibrium temperature outlined in SM
F 3). In other words, we see that the criticality is self-
organizing (SOC) through the entire simulation, and
it avoids critical slowing down at all points in time.
should be noted that SOC is not an intrinsic property of frustrated short-ranged spin glasses, and this is evident in Fig. 4 of the SM, which displays a persistently hypercritical CSD for stochastic clusters generated by the Swendsen-Wang (SW) and ICM rules. This also confirms the inefficiency of traditional cluster algorithms for frustrated systems.

Note that the SOC behavior of the memory induced clusters is verified for multiple 2d and 3d finite-ranged spin glasses (planted or not) with a few 3d examples (EA, PF, and FF models) given in Fig. 5 of the SM. There is strong evidence that the Fisher exponent of the CSD is only dependent on the frustration ratio of the underlying glass, but independent of the effective temperature or lattice size. We encourage the readers to experiment with glasses in higher dimensions and other connectivity structures using the available codes.

**IV. FINDING A GLASSY GROUND STATE**

Finally, we show that the aforementioned non-equilibrium critical behavior allows us to find the ground state of spin glasses efficiently. First of all, we note that below the critical temperature $T_c$, the energy landscape of a spin glass becomes highly non-convex, and most algorithms fail to efficiently navigate it. For "artificial" glassy instances where the finite residual entropy is suppressed via the coupling of local interaction states, the inefficiency of these simulations is exposed most prominently when the temperature is lowered to the $T = 0$ limit. To benchmark the efficiency of an algorithm for glass simulations, one can record the relaxation time, or equivalently the time-to-solution (TTS) to the ground state over a sample of glass realizations. We compare the memory dynamics against simulated annealing (SA) (see SM C1) and ICM (see SM C3). Note that ICM is the best known replica-based algorithm for simulating spin glasses in any dimension. See Section C and D in SM for detailed discussions on how the TTS can be fairly measured for different algorithms in terms of the number of sweeps.

With this goal in mind, we use a class of 3d glass instances where the frustration ratio can be controlled (see SM F2 for an explanation of how these instances are generated). Since these instances are planted, the ground state energies are known in advance. This way we can verify the correctness of the algorithm. To perform the evaluation, we generate fully-frustrated 3d glasses up to size $8 \times 8 \times 10$, and partially-frustrated 3d glasses up to size $32 \times 32 \times 32$ (see SM F2 for details), with 400 randomly generated instances per size. For each size, we evaluate the efficiency of every algorithm by collecting its sweep statistics up to the 60-th percentile (see SM F1), and estimating the scaling behavior of the median number of sweeps. The implementation used for the scalability test is detailed in SM D.

To dispel beliefs that we are fine-tuning the parameters of the memory dynamics just to solve planted benchmarks, we perform the test also on the prototypical EA model for baseline reference, without changing the parameters. Since the EA model is not planted, it is exponentially hard to verify correctness of the algorithms. Instead, we use a fixed sample of 400 random realizations of $20 \times 20 \times 20$ EA models for all algorithms, and monitor the log-deviation of the best Ising energies found so far (above the best known ground state energy) throughout the simulation (see SM E).

As shown in Fig. 3, for the fully-frustrated instances, while the scaling of standard stochastic algorithms are well-fitted by super-polynomial functions (an exponential for SA and a sub-exponential for ICM), the scaling of the memory dynamics is well-fitted by a polynomial up to the maximum size we have tested. For partially-frustrated instances, all three algorithms appear to scale polynomially, with the memory dynamics having the lowest power. For the EA instances, the returned energy of the memory dynamics is mostly below the other two algorithms, and appears to continue evolving asymptotically at a lower energy as well.

In a word, not only is the memory dynamics more efficient in navigating the glassy landscapes at low temperature, but it is also more efficient in discovering "deep" solutions. Most importantly, its efficiency has been empirically proven to be general on complex spin glasses. Nevertheless, one should note that the algorithm operates at non-equilibrium (unlike SA and ICM), so at the present stage, it is not clear how equilibrium statistics can be efficiently sampled. This is a work in progress.

**V. CONCLUSIONS**

In this work, we have introduced a new approach to simulate spin glasses based on the coupling of (linearly relaxed) spins with memory variables. We have shown numerically that the generated memory-induced spin clusters are critical, and the memory dynamics are efficient in finding the ground state of fully-frustrated Ising spin glasses in 3d, even using the basic forward Euler dis-
FIG. 3: (Top) Scalability of the median number of sweeps for the fully- and partially-frustrated 3d spin glass (FF and PF) for simulated annealing, parallel tempering with isoenergetic cluster move, and memory dynamics (SA, ICM, and Mem), measured as the total number of sweeps (see SM B for justification) on a log-log scale. Statistics are collected over 400 runs, and the shaded region denotes the 40-th to 60-th percentile, with the fitting done with log-linear regression. The estimated scaling constants for SA, ICM, and memory are $\{0.027 \pm 0.0005, 0.67 \pm 0.03, 4.1 \pm 0.1\}$ for FF and $\{0.98 \pm 0.01, 0.96 \pm 0.05, 0.79 \pm 0.02\}$ for PF, as noted in the legends. The fitting is done on the last 6 data points for every algorithm. (Bottom) The deviation of the best mean Ising energy from the expected ground state found over a fixed sample of 400 3d Edwards-Anderson (EA) glass realizations, monitored in simulation time (sweeps). The energy returned by the memory dynamics is mostly below the other two throughout the simulation. Note that since the EA model is not planted, it cannot be verified whether the true ground state of the sample is reached (see SM E for further discussion).

cretization scheme. As a future development, the introduction of an appropriate discretization and acceptance scheme may endow the memory dynamics with the detailed-balance property, making it applicable to simulating equilibrium dynamics of glasses even at finite temperature. This would allow the algorithm to be interfaced with modern stochastic algorithms, which may be useful for generating critical clusters for bosonic quantum spin and gauge systems which has been shown empirically in the past.

VI. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 2034558. Y.P. would like to thank Firas Hamze for stimulating discussions on the entropic properties of the tiling cubes, and Zheng Zhu for clarifying the implementation details of the ICM algorithm. All the numerical results presented in this study have been done on a single core of an AMD EPYC server. They can be reproduced using the codes in the repository PeaBrane/Ising-Simulation.
The repository is a complete suite for Ising Simulation in MATLAB that is maintained and developed by Y.P.

1. W. Barthel, A. K. Hartmann, M. Leone, F. Ricci-Tersenghi, M. Weigt, and R. Zecchina, Physical review letters 88, 188701 (2002).
2. A. Fischer and C. Igel, in Iberoamerican congress on pattern recognition (Springer, 2012) pp. 14–36.
3. W. Bialek, A. Cavagna, I. Giardina, T. Mora, E. Silvestri, M. Viale, and A. M. Walczak, Proceedings of the National Academy of Sciences 109, 4786 (2012).
4. S. F. Edwards and P. W. Anderson, Journal of Physics F: Metal Physics 5, 965 (1975).
5. E. Ising, Zeitschrift für Physik 31, 253 (1925).
6. M. Mézard, G. Parisi, and M. Virasoro, Spin glass theory and beyond: An Introduction to the Replica Method and Its Applications, Vol. 9 (World Scientific Publishing Company, 1987).
7. T. Castellani and A. Cavagna, Journal of Statistical Mechanics: Theory and Experiment 2005, P05012 (2005).
8. L. Onsager, Physical Review 65, 117 (1944).
9. P. W. Kasteleyn, Physica 27, 1209 (1961).
10. J. Edmonds, Journal of Research of the National Bureau of Standards B 71, 233 (1967).
11. F. Barahona, Journal of Physics A: Mathematical and General 15, 3241 (1982).
12. S. Istrail, in Proceedings of the thirty-second annual ACM symposium on Theory of computing (2000) pp. 87–96.
13. A. H. Teller, E. Teller, A. Coniglio, Physical Review E 54, 394 (1996).
14. B. Selman and H. Kautz, in IJCAI, Vol. 93 (Citeseer, 1993) pp. 290–295.
15. R. J. Glauber, Journal of mathematical physics 4, 294 (1963).
16. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, science 220, 671 (1983).
17. H. Suwa and S. Todo, Physical review letters 105, 120603 (2010).
18. Y. Iba, Transactions of the Japanese Society for Artificial Intelligence 16, 279 (2001).
19. B. Selman and H. Kautz, in IJCAI, Vol. 93 (Citeseer, 1993) pp. 290–295.
20. S. Boettcher and A. G. Percus, in Computational Modeling and Problem Solving in the Networked World (Springer, 2003) pp. 61–77.
21. J.-C. Walter and G. Barkema, Physica A: Statistical Mechanics and its Applications 418, 78 (2015).
22. C. M. Fortuin and P. W. Kasteleyn, Physica 57, 536 (1972).
23. R. H. Swendsen and J.-S. Wang, Physical review letters 58, 86 (1987).
24. U. Wolff, Physical Review Letters 62, 361 (1989).
25. F. Niedermayer, Physical review letters 61, 2026 (1988).
26. D. Kandel, R. Ben-Av, and E. Domany, Physical Review B 45, 4700 (1992).
27. P. Coddington and L. Han, Physical Review B 50, 3058 (1994).
28. Y. R. Pei and M. Di Ventra, In preparation.
29. E. De Santis and A. Gandolfi, Annals of probability, 1781 (1999).
30. J. Houdayer and A. K. Hartmann, Physical Review B 70, 014418 (2004).
31. Z. Zhu, A. J. Ochoa, and H. G. Katzgraber, Physical review letters 115, 077201 (2015).
32. E. Marinari and G. Parisi, EPL (Europhysics Letters) 19, 451 (1992).
33. A. Morningstar and R. G. Melko, The Journal of Machine Learning Research 18, 5975 (2017).
34. L. Wang, Physical Review E 96, 051301 (2017).
35. N. Le Roux and Y. Bengio, Neural computation 20, 1631 (2008).
36. M. Di Ventra and Y. V. Pershin, Nature Physics 9, 200 (2013).
37. M. Di Ventra and F. L. Traversa, J. Appl. Phys. 123, 180901 (2018).
38. F. Sheldon, F. L. Traversa, and M. Di Ventra, Physical Review E 100, 053311 (2019).
39. S. R. Bearden, Y. R. Pei, and M. Di Ventra, Scientific reports 10, 1 (2020).
40. H. Manukian, Y. R. Pei, S. R. Bearden, and M. Di Ventra, Communications Physics 3, 1 (2020).
41. M. X. Goemans and D. P. Williamson, Journal of the ACM (JACM) 42, 115 (1995).
42. F. Hamze, D. C. Jacob, A. J. Ochoa, D. Perera, W. Wang, and H. G. Katzgraber, Physical Review E 97, 043303 (2018).
43. D. Sherrington and S. Kirkpatrick, Physical review letters 35, 1792 (1975).
44. M. Kac and C. J. Thompson, Physica Norvegica 5, 163 (1971).
45. Y. Pei, github/PeaBrane (2020).
46. F. L. Traversa and M. Di Ventra, Chaos: An Interdisciplinary Journal of Nonlinear Science 27, 023107 (2017).
47. S. Zhang and A. G. Constantinides, IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing 39, 441 (1992).
48. V. Cataudella, G. Franzese, M. Nicodemi, A. Scala, and A. Coniglio, Physical Review E 54, 175 (1996).
49. J. Villain, R. Bidaux, J.-P. Carton, and R. Conte, Journal de Physique 41, 1263 (1980).
50. J. Besag, J. Roy. Statist. Soc. Ser. B 56, 591 (1994).
51. M. Betancourt, arXiv preprint arXiv:1701.02434 (2017).
52. J. S. R. Bulirsch, Introduction to Numerical Analysis (Springer, 2010).
53. Y.-H. Zhang and M. Di Ventra, arXiv preprint arXiv:2102.03547 (2021).
54. M. Hassan and M. Rahman, Physical Review E 92, 040101 (2015).
55. A. A. Saberi, Physics Reports 578, 1 (2015).
56. P. Bak, C. Tang, and K. Wiesenfeld, Physical review letters 59, 381 (1987).
57. J. Hesse and T. Gross, Frontiers in systems neuroscience 8, 166 (2014).
58. E. Marinari, G. Parisi, and F. Ritort, Journal of Physics A: Mathematical and General 28, 327 (1995).
59. J. C. Andreassen, Z. Zhu, R. S. Andrist, H. G. Katzgraber, V. Dobrosavljević, and G. T. Zimanyi, Physical review letters 111, 097203 (2013).
C. R. Rao, in *Breakthroughs in statistics* (Springer, 1992) pp. 235–247.

K. Binder, *Zeitschrift für Physik B Condensed Matter* **43**, 119 (1981).

G. Parisi, *Physical Review Letters* **50**, 1946 (1983).

E. Marinari, G. Parisi, F. Ricci-Tersenghi, J. J. Ruiz-Lorenzo, and F. Zuliani, *Journal of Statistical Physics* **98**, 973 (2000).
SM A: Critical Percolation

Very briefly, percolation is a random process on a graph where a bond is opened (or a site is “occupied”) with a given probability \( p \), and this usually generates multiple clusters on the graph (for now assume the graph is a lattice) connected by open bonds\(^{55}\). For most graphs, there is a percolation threshold \( p_c \) such that when \( p < p_c \), all the clusters are finite, and when \( p > p_c \), there is a unique giant cluster spanning a constant fraction of the lattice. One of the most important characterizations of the percolation process is the finite cluster size distribution (CSD), \( n(s) \), which counts the number of clusters of a given size \( s \) (excluding the single infinite cluster). In both the subcritical and supercritical regime \((p \neq p_c)\), \( n(s) \) decays exponentially with respect to \( s \), while at criticality \((p = p_c)\), \( n(s) \sim s^{-\gamma} \) decays as a power law with the critical exponent \( \gamma \) referred to as the Fischer exponent, which is one of the many scale-free properties of critical percolation\(^{55}\). The majority of spin models can be translated into a modified percolation process\(^{22}\), which has been the inspiration of many cluster algorithms over the past few decades\(^{24,26,30}\). In another work we have analyzed the efficiency of these cluster algorithms both analytically and empirically\(^{28}\).

Both the ICM algorithm (see Section C 3) and memory dynamics (see Section D) can be naturally studied from a percolation perspective. During the Houdayer cluster formation in the ICM algorithm\(^{40}\), spin sites with negative overlap between a replica pair can be interpreted as occupied sites, and sites with positive overlap are unoccupied. Randomness is introduced into the system in the form of thermalization\(^{28}\) generated by Metropolis sweeps and replica exchanges. There are certain procedures ensuring that the largest cluster size does not span the entire lattice. First, the cluster move only occurs between replica pairs of sufficiently low temperature, and second, whenever the number of negative sites exceeds half the spins, one of the replica is flipped globally to suppress the percolation process. However, despite these restrictions, it is shown that the algorithm still fails to be efficient in general, as it is heavily reliant on the underlying graph structure\(^{28,31}\).

Unlike the ICM algorithm, the percolation process defined by the memory variables we have introduced in the main text is a bond percolation process. However, the more important distinction is that the memory variables are continuous, meaning that they naturally induce a weighted graph, where each edge weight denotes the percolation probability on the bond. It has been suggested that a weighted lattice may display different critical properties than the unweighted counterparts\(^{54}\). In this work, we update the distribution \( n(s) \) at each simulation time unit (instead of each adaptive time step to ensure efficient and unbiased sampling\(^{70}\)). We find that the distribution \( n(s) \) follows a power-law decay with the giant component being absent. This suggests that the memory induced percolation process is near criticality, so that the memory dynamics are efficient in sampling the underlying glass near \( T_c \)\(^{48,71}\).

To make practical use of the clusters generated by the memory variables, we can perform a Swendsen-Wang (SW) update\(^{23}\) on these clusters as an intelligent restart method in a digital implementation of the memory dynamics. The SW update entails flipping each cluster independently with probability \( \frac{1}{2} \). Note that this method satisfies detailed balance even if the percolation ratios defined by the memory variables are not uniform across the graph. Alternatively, one can also opt to perform a Wolff update\(^{24}\) which involves flipping one randomly chosen cluster, though the two methods are similar in efficiency when the CSD is critical. Note that this algorithmic step is not central to the efficiency of the memory dynamics, though it does help to increase the TTS by a small constant factor.

For stochastic clusters generated using algorithmic bond-formation rules, such as Swendsen-Wang\(^{23}\) or Houdayer\(^{30}\) clusters, the CSD tends to be hyper-critical for frustrated models\(^{27}\), due to the mismatch between the critical temperature \( T_c \) and the critical threshold for the effective (bond- or site-) percolation ratio \( p_c \).\(^{48}\) In the SW case, this can be expressed as

\[
T_c \ll \frac{2}{\log (1/(1 - p_c))},
\]

meaning that the CSD is already critical far above the critical temperature. Similar expressions can be derived for the Houdayer clusters as well. A study focusing on the inefficiency of stochastic clusters is detailed in another work\(^{24,26}\), and we here simply show that the empirical CSD for SW and Houdayer rules on the fully-frustrated 3d Ising glass is persistently hyper-critical, as shown in Fig. 4, meaning that cluster algorithms employing such non-local update rules cannot be generally efficient, especially if the frustration ratio of the underlying glass is large. On the other hand, the clusters generated by the memory variables are critical at any time during evolution regardless of the underlying
FIG. 4: The cluster size distribution (CSD) for the Swendsen-Wang (SW) and Houdayer percolation rules on a fully frustrated 3d Ising glass sized $10 \times 10 \times 10$. The equilibrium statistics is collected over 100 realizations of disorder, simulated with parallel tempering (PT) over $2^{15}$ sweeps with a waiting time of $2^{19}$. Note that both classes of clusters are hypercritical at the critical temperature $T_c \approx 1.20$ (see Section F 4). While the SW clusters become increasingly hypercritical as the temperature is lowered, the ICM clusters appear to be hypercritical at all temperatures, even when the largest component is restricted to half the lattice size (see Section C 3).

frustration profile of the glass, though the Fisher exponent seems to depend on the frustration ratio of the underlying glass model (see Fig. 5).

SM B: Complexity

Practically speaking, the most direct measure of complexity is the wall time required to run a given algorithm until the solution is reached. However, this measure of complexity suffers from the uncertainty due to a number of irrelevant variables that are hard to measure or control, originating mainly from not only the details of the algorithmic design, but also the programming language implementation and the hardware architecture. For example (as further discussed in SM D), the efficiency of simulating the ODEs for memory dynamics will depend on how well the code is vectorized and (further downstream) how the SIMD instructions are handled by the CPU, none of which are directly controllable nor relevant to the time complexity of the algorithm (though they have been optimized to the extent of the authors’ ability in order to push the scalability test as far as possible).

From a theoretical standpoint, the main focus is the time and space complexity of the algorithm, which can be roughly interpreted as the scaling of the computational cost and memory requirement of the TTS with respect to the size of the problem. As the scalability is the main concern here, the actual time and memory (RAM) required to run the algorithm for a specific problem type is not of major interest, meaning that any complexity measurement and algorithmic implementation that differ in the prefactor or additional terms of smaller powers should be treated as equivalent. For example, if the time required to run an algorithm is $aN^3 + bN^2$, where $N$ is the size of the problem, then the actual prefactor $a$ and the entire second term $bN^2$ are inconsequential to the time complexity, which is $O(N^3)$. Finally, since it is clear that SA, ICM, and memory dynamics should scale linearly with respect to memory (RAM) requirements, we will not be focusing on that here.

The only measure of interest should then be the time complexity of both algorithms running the same class of

---

5 This is because only the spin states are stored in the memory during the simulation, and a sweep is essentially a one-pass algorithm that does not create any new array structures. For replica-based algorithms like ICM, the number of replicas are constant with respect to system size (see Section C 3), so they only contribute a pre-factor to the space complexity.
FIG. 5: The clusters generated by the memory variables appear to be near criticality at any time during evolution on the fully-frustrated, partially-frustrated, and Edwards-Anderson (FF, PF, and EA) 3d Ising glass, with the decay power seeming to depend only on the frustration ratio of the underlying glass (a higher frustration yields a lower power). The equilibrium statistics is collected over 400 realizations of disorder and a time window of $\Delta t = 2^6$ beginning at simulation times $t_0 = \{0, 1300, 4200\}$.

instances. For both algorithms, we estimate the time complexity numerically using the median statistics of TTS over 400 tiling glass instances\textsuperscript{42}, with the 40-th to 60-th percentiles reported to verify the robustness of the algorithm. The reason we chose to record the median instead of the mean\textsuperscript{6} is because the hardness of the instances follows roughly a log-normal distribution\textsuperscript{42}. One can always extrapolate the scaling of the mean TTS by doing regression on the reported median and the percentile statistics, by using a log-linear model\textsuperscript{76}. For measurement of TTS, one can measure (or estimate) the number of FLOPS (floating-point operations) required for the algorithm to find the solution. Here, we report an equivalent measure of the number of sweeps over the lattice (see Section C and D). Since a sweep is an one-pass algorithm, the number of FLOPS can be evaluated by simply multiplying the number of sweeps with the system size (number of spins), which adds a linear power to the scalability. Note that this will not change whether the time complexity of an algorithm is polynomial or super-polynomial, or the efficiency comparison between different algorithms in general. For readers interested in scalability of FLOPS, we encourage them to re-implement the algorithms in their language of choice referring to our own MATLAB implementation\textsuperscript{45}, which unfortunately is not streamlined for the measurement of FLOPS without sacrificing too much practical efficiency.

\textsuperscript{6} There is a debate on whether one should measure the median or mean for empirical studies of scalability. Practically speaking, measuring the median is much more computationally inexpensive because it requires only solving up to the 50 percentile of the sample in terms of TTS. However, the mean TTS is of more theoretical interest, because the definition of self-reducibility is based on whether the mean complexity equates to the worst-case complexity.
SM C: Implementation of Stochastic Algorithms

Regardless of how efficient the employed cluster update routine is, all stochastic algorithms inevitably use an ergodic routine referred to as a sweep, where all the spins in the lattice are updated sequentially in a Metropolis-type acceptance scheme. To be more precise, whenever a single spin $\sigma_i$ in the lattice is flipped (from $\sigma_i$ to $-\sigma_i$), the change in the Ising energy associated with it is

$$\Delta E_i = 2 \sum_{ij} J_{ij} \sigma_i,$$

and the Metropolis acceptance ratio of this update is then given as

$$P(\sigma_i \rightarrow -\sigma_i) = \min \left(1, \exp \left(-2\beta \Delta E_i \right) \right).$$

A single sweep usually occurs at a constant inverse temperature $\beta$, making it generally interfaceable with a plethora of cluster or replica algorithms.

In most cases, it is important for the spins to be updated sequentially, as many attempts of trying to introduce synchronous update methods (such as stripe-wise updates) generally lose more in ergodicity than gain in parallel efficiency. This means that in a computational implementation, a single sweep is best limited to a single core, and it usually constitutes the most computationally intensive routine when used in conjunction with cluster updates. The complexity is simply $O(N)$, where $N$ is the number of spins, noting that the coordination number of the 3d lattice is fixed at $z = 6$. Several modern algorithms that efficiently utilize the sweep routine include simulated annealing (SA), parallel tempering (PT), and isoenergetic cluster moves (ICM), with their scalabilities of the median TTS for the fully frustrated 3d Ising glass shown in Fig. 3 of the main text (with the PT and ICM algorithms combined).

### 1. Simulated Annealing

The Simulated Annealing (SA) algorithm is inspired by the physical annealing process in metallurgy, where the metal is gradually cooled from a high temperature to a low one to improve its ductility. In the context of optimization, this means that we begin with a high temperature, and gradually lower the temperature to a very small value, and this process somewhat aids the algorithm in navigating the non-convex cost function of the optimization problem to find the global minimum. This algorithm saw great success in many industrial optimization problems, such as the traveling salesman problem (TSP). Many state-of-the-art algorithms are based on the same underlying concept, with added entropic routines for intelligent exploration of the state space. Here, we are using the algorithm in its original form, with the annealing parameters extensively optimized.

In most cases, a single run of an SA routine is not sufficient to find the ground state, and more often than not, a restart routine is generally required, to give the algorithms multiple chances at tackling the problem. Although there have been studies on using informed restart methods to interface with the algorithm, these methods are not general, and usually only provide a pre-factorial improvement over a random restart routine. Therefore, in this work, we will simply use the random restart routine to minimize complications.

The three important parameters for SA are $\beta_{\text{min}}$, $\beta_{\text{max}}$, and $t_{\text{sweep}}$, together referred to as the annealing routine. $\beta_{\text{min}}$ is the starting inverse temperature of the sweep, $\beta_{\text{max}}$ is the ending inverse temperature of the sweep, and $t_{\text{sweep}}$ is the number of sweeps going from $\beta_{\text{min}}$ to $\beta_{\text{max}}$. Based on seminal work, and extensive optimization studies done previously, we find that the best annealing routine of $\beta$ is linear from $\beta_{\text{min}} = 0.1$ to $\beta_{\text{max}} = \log(N)$, where $N$ is the number of spins. Furthermore, we find that the optimal number of sweeps between restarts is $t_{\text{sweep}} = N$ regardless of the underlying frustration profile of the glass, which balances between having a sufficiently gradual annealing ratio and sufficient restart opportunities. The annealing routine can be expressed succinctly as

$$\beta = 0.1 + \left(\log(N) - 0.1\right) \frac{t - 1}{t_{\text{sweep}}},$$

where the inverse temperature is increased from 0.1 to $\log(N)$ over $N$ sweeps.
2. Parallel Tempering

As mentioned in the previous section, the major problem with using SA is the lack of a generally intelligent restart method, so in practice, most practitioners simply use a random restart routine, where all the spins in the lattice are uniformly sampled from $\sigma_i = \pm 1$. As a substantial improvement, parallel tempering (PT) replicates the lattice into multiple copies, and simulates the replicas under different temperatures\textsuperscript{32} (usually by sweeping the lattice), and two replicas of neighboring temperatures are exchanged depending on an external rule to improve efficiency of exploring the Gibbs measure. The acceptance ratio of the exchange is given by

$$P(\sigma^a \leftrightarrow \sigma^b) = \min \left( 1, \exp \left( (\beta^a - \beta^b)(E(\sigma^a) - E(\sigma^b)) \right) \right),$$

which keeps the joint distribution of the entire replicated system stationary. Intuitively, the replica at the highest temperature is essentially sampling from the uniform spin measure, and this “random restart” propagates down the replica chain through the exchange interactions, where multiple replicas essentially “mediate” the restart routine from $\beta_{\text{min}}$ to $\beta_{\text{max}}$. This is the reason why PT is often times considered as an algorithm with an implicit restart routine that is “intelligent”. This is arguably the most general algorithm designed to work for many classes of optimization problems on different underlying graph structures\textsuperscript{42,60,85}, including many industrial problems\textsuperscript{86,87}. Combined with intra- or inter-replica cluster algorithms\textsuperscript{24,26,30}, this results in incredibly efficient stochastic algorithms, where non-local updates are complemented by an intelligent restart method. We will discuss one such algorithm in the next section. In this work, we use $n_r = 30$ replicas spaced geometrically in inverse temperature from $\beta_{\text{min}} = 0.1$ to $\beta_{\text{max}} = \log(N)$, or

$$\beta_i = \beta_{\text{min}} \left( \frac{\beta_{\text{max}}}{\beta_{\text{min}}} \right)^{(i-1)/(n_r-1)},$$

based partially on existing work\textsuperscript{38,42} and our own optimization attempts. Note that the exchange update is trivial in computational cost, as it simply involves computing an acceptance ratio and exchanging the indices of two replicas.

3. Isoenergetic Cluster Moves

The ICM algorithm is currently the state-of-the-art cluster algorithm\textsuperscript{31} for simulating spin glasses that combines the method of PT\textsuperscript{32} and Houdayer cluster updates\textsuperscript{30}. In the main text, this is then used as the most representative of stochastic algorithms to compare against the memory dynamics in the TTS scaling behavior on the fully-frustrated 3d Ising model\textsuperscript{42}. A comprehensive description and the pseudocode for the algorithm is already given in the literature\textsuperscript{31}, and also implemented in MATLAB\textsuperscript{45}. Here, we provide a brief overview of the algorithm and list the parameters that we used to perform the simulations. In addition, we pinpoint the most computationally intensive routine, whose scalability we use as the time complexity of the algorithm.

In this algorithm, the parallel tempering algorithm\textsuperscript{32} is coupled with Houdayer cluster moves\textsuperscript{30}. To begin, a number of replica pairs are initialized, with the pairs spaced geometrically in temperature. After one sweep in every replica, an Houdayer update is performed for every replica pair. This cluster update flips a non-trivial cluster of spins with negative overlap between two replicas to increase the mixing rate. Finally, the parallel tempering routine attempts to exchange the temperature between two random replicas of neighboring temperatures, to improve the thermalization process. This algorithm can be used both as a sampler and an optimizer. For the latter case, one simply has to record and return the lowest energy that is sampled by the algorithm. We use the same parameters as given in Section C2, meaning that the total number of replicas is $2n_r = 60$.

Note that in the modern implementation of the ICM algorithm\textsuperscript{31}, there is an extra algorithmic step that performs a global spin flip on a replica in each pair, whenever the number of negative overlap sites exceeds half the lattice size, so that the largest cluster component never exceeds half of the lattice. Although the intention of this procedure is to suppress the percolation process through restricting the size of the giant component, similar to the intention of plaquette-based bond-formation rules\textsuperscript{26}, it does not fundamentally address the issue of mismatching the critical temperature and percolation threshold (as shown in Fig. 4 in the SM), meaning that Houdayer clusters are still hyper-critical despite algorithmic interventions. It is also important to note that the global spin-flip routine breaks detailed balance, because the reverse transition probability is zero, meaning that the global flip should in theory never be accepted. Further discussion of this phenomenon is given in another work\textsuperscript{28}. 

\[ \sigma_i = \pm 1 \]
To analyze the computational complexity of the different routines to inform a fair TTS measure, we note that the PT routine is trivial in cost (as noted in Section C 2), and in the worst-case, the Houdayer move performs either a breadth-first or depth-first search (BFS or DFS) to identify the spin clusters, whose time complexity is $O(N)$ for the lattice graph. This is also the time complexity of one Metropolis sweep over a replica. Therefore, the complexity can be measured as the total number of sweeps summed over all the replicas, with the Houdayer update cost “generously” ignored/absorbed into the total complexity as it is of the same complexity power.

SM D: Integration of Memory Dynamics

Recall that the memory dynamics is formulated in continuous time, and originally conceived to be implemented on physical circuits. However, it was later discovered that a carefully designed memory system is robust against noise and perturbations, and can be readily simulated numerically on a digital computer using basic integration schemes such as forward Euler. Such a basic implementation has been proven to perform exceptionally well for multiple problem structures, as long as the relative timescale of the memory variables (with respect to the spins) is appropriate. Of course, this leaves room for many improvements in numerical methods to speed up the rate of convergence of the dynamics when the goal is to implement the memory dynamics digitally as a practical solver. Here, we present a few improvements that we found relevant to our work. First of all, we rewrite Eq. (5) in the main text for convenience of the reader, where \( \{\alpha, \beta, \gamma, \delta, \zeta\} = \{0.80, 1.04, 1.67, 7.07, 2.20\} \) are constants chosen for the 3d cubic graph, and are fixed for system sizes and the type of glass (as long as \( |J_{ij}| = 1 \)).

\[
\begin{align*}
\dot{x}_{ij} &= \gamma C_{ij} - y_{ij} \\
\dot{y}_{ij} &= \delta x_{ij} - \zeta, \\
\text{where} \quad C_{ij} &= \frac{1}{2}(J_{ij} \sigma_i \sigma_j + 1) \in [0, 1].
\end{align*}
\] (D1)

The function \( C_{ij}(\sigma) \) is referred to as the clause function in the field of constrained optimization, evaluating to +1 if the spin interaction is satisfied, and 0 otherwise. Note that we use \( C_{ij} \) purely to notationally interface with the literature, and the offset from \( J_{ij} \sigma_i \sigma_j \) can be easily accounted for by a constant shift in the initialization of \( y \). This set of equations and parameters is used to perform the simulation of the CSD as presented in Fig. 2 in the main text.

Typically, to ensure positivity of \( x \), we would opt for an exponential growth rate given as \( \dot{x}_{ij} = (\gamma C_{ij} - y_{ij}) x_{ij} \), which in fact already outperforms stochastic algorithms in TTS. Nevertheless, we choose to make the growth of \( x \) linear for faster dynamics, and to guarantee that no “hidden exponential” is present during dynamics. However, this comes at the price of potential negativity of \( x \) and the introduction of unstable modes. To avoid this, we can dynamically anneal the decay rate via the extra (long-term) memory variable \( y \) coupled bond-wise to \( x \). Intuitively, the new memory variable \( y \) grows/decays along with \( x \) with some time lag to ensure that the relative change in the magnitude of \( x \) is never too large nor too small. Though not necessary in most cases, positivity of these memory variables can be simply enforced by introducing explicit bounding values at each time step as such,

\[
\begin{align*}
x_{ij,n+1} &= \min\{\max\{x_{ij,n} + dt (\gamma C_{ij} - y_{ij}), 0\}, 1\} \\
y_{ij,n+1} &= \min\{\max\{y_{ij,n} + dt (\delta x_{ij} - \zeta), 1\}, 10\},
\end{align*}
\]

In most cases, the variable \( y \) is only relevant to the initial transient dynamics in its purpose of suppressing the highly oscillatory memory modes (when the spin glass is far from equilibrium, a rapid relaxation of the spins induces large fluctuations in the magnitude of \( x \)). When the system relaxes slightly, the variable \( y \) will decay quickly to its lower bound (if the parameter \( \zeta \) is chosen appropriately), and will effectively serve as a constant decay of −1 for \( x \). If the memory dynamics dive quickly below \( c \) (see Fig. 2 in the main text), then \( y \) is usually not needed, but we include it here for the sake of generality. A formal analysis of this discretization/bounding in the context of stability, absence of periodic orbits and chaos is given in the supplementary material of our previous work. Note that

---

7 In the context of our study, robustness means that multiple trajectories emanating from any initial point in the phase space will go to the optimum. Therefore, it is not necessary for us to accurately integrate any particular one of them. It is possible for us to end up in another “desirable” trajectory after deviation from the original one, and still find the optimum in the end.
we do not leverage any stochasticity in taming discretization errors\cite{50,51}, which is further empirical proof of robustness.

Besides the memory variables, the step size itself can be also made adaptive\cite{93} to further improve stability and speed up convergence. We adapt the step size as such,

$$dt_n = \min \left\{ \frac{1}{\max_i(|\dot{x}_i|)}, 2^{-5}, 2^{-3} \right\},$$

to regularize the maximum voltage change at every step\footnote{Note that this time step adaptive schedule we choose to use is rather unconventional. The standard adaptive schedule is to geometrically tune the step size based on the local error estimate for the purpose of efficiently simulating the solution trajectory. However, in our case, we do not require such accuracy, and employing such procedure actually decreases the rate of convergence to the optimum.}. Although not necessary, an intelligent restart method\cite{82} can also be implemented to ensure that the energy landscape is being thoroughly explored by the memory dynamics. After initialization, the dynamics are integrated up to time $t_0 = 2^6$, and if the optimum is not found in the time duration, then we perform a SW update on clusters generated by $x$ (see Section A), and continue to run the dynamics for another iteration, until the ground state is discovered or the total timeout is reached.

Note that the time complexity of performing an integration step is $O(N)$, which is the same as a single sweep in stochastic algorithms (see Section C). However, when implemented in hardware, integrating a time-step is always much faster than a sweep, because the spin updates in our integration scheme are synchronous (as with standard explicit methods for simulating multivariate ODEs), meaning that the machine code can be vectorized to interface with a single instruction, multiple data (SIMD) hardware structure\cite{72}, whereas it would be incredibly difficult to do so with a sweep, even if the spin data are stored bit-wise. Again, to be generous, we do not consider such hardware overhead in TTS measures, and simply measure the TTS as the total simulation time. Note that since the adaptive time step is bounded below by a constant $dt_{\min} = 2^{-5}$, there is no cost associated with the inverse scaling of step size with respect to $N$\cite{89}. In the end, the redundant scaling $O(N)$ is factorized out for both the TTS measures for stochastic algorithms and the integration of memory dynamics, as it only affects the polynomial order of the TTS scaling, but it does not affect whether the scaling is polynomial or exponential.

Furthermore, we note that the parameters $\{\alpha, \beta, \gamma, \delta, \zeta\}$ for the memory dynamics were tuned very minimally using simplex descent\cite{94}, with the cost function being the mean Ising energy returned for 40 runs of memory dynamics on the $6 \times 6 \times 6$ uniform random-bond Ising glass\cite{95}. This ensures that we are not taking advantage of the specific structure of the tiling instances by over-fitting the parameters specifically for the fully frustrated 3d Ising glass. This is similar to the technique used in machine learning for tuning hyperparameters such as the learning rates\cite{96}, where to avoid over-fitting, the parameters are tuned on some given neural network (NN) for a subset of tasks that it is designed for and then the NN is tested on another disjoint subset of tasks for cross validation of performance\cite{97}. On the other hand, the parameters for SA and ICM (annealing schedule and temperature spacing respectively) are tuned extensively based on previous works\cite{31,38}, which effectively tilts the playing field in favor of stochastic algorithms.

### 1. Non-local Extensions

The main culprit behind the inefficiency of local cluster algorithms is that the bond-formation rules are defined edge-wise, such as the SW rule, so they completely ignore the non-local effects of frustration. Therefore, such algorithms are prone to over-percolate\cite{28,48,71}, and are generally inefficient for frustrated systems. This has inspired multiple extensions of the original SW rule to more non-local unit cells. For instance, the fully frustrated Ising model (FFIM) in $2d$\cite{19} can be deconstructed into unit cells of checkered plaquettes, with each plaquette guaranteeing exactly one negative interaction at the ground state. This led to the realization that bond-formation decisions can be made plaquette wise, resulting in the prototypical KBD plaquette rules\cite{26}. This in turn inspired a plethora of other non-local cluster algorithms\cite{38,62} designed for other classical and quantum glasses. The KBD plaquette rule has proven to be efficient in reducing the auto-correlation time in simulating magnetization properties\cite{27}, though its efficiency is still largely restricted to the $2d$ FFIM\cite{28}.

The equations for the memory dynamics (see Eq. (D1)) can also be modified to interface with such non-local bond-formation rules. For instance, if we take the $2d$ FFIM, and index the checkered plaquettes (or checkered cubes in 3D)
FIG. 6: The green line is the fitted scalability for the median TTS for the bond-based long term memory (LTM) on the fully-frustrated 3d glass, and the orange line is for the checkered LTM on the same instances. The scaling powers are $4.4 \pm 0.1$ and $4.1 \pm 0.1$ respectively, with the checkered based LTM displaying slightly favorable power. Note that the medium TTS’s are well-fitted by polynomial functions for both schemes.

as $\square$, we can restrict the long-term memory variables to these plaquettes, and couple it to the short-term memory $x$ as, e.g.,

$$y_{\square} = \delta \sum_{(i,j) \in \square} x_{ij} - \zeta.$$  

Instead of simply regularizing $x$ as discussed in Section D, the long-term memory now provides additional non-local information to the original edge coupled system of $\{\sigma, x\}$. To make the analogy with machine learning, we note that the use of multiple layered network structures is common for pattern recognition in deep learning, and here, $y$ acts as an additional layer of nodes, learning to recognize plaquette frustration patterns. To take this analogy further, we could introduce more layers of memory to learn successively non-local cell patterns, as long as stability is still guaranteed. For this implementation, we use a slightly different set of parameters $\{\alpha, \beta, \gamma, \delta, \zeta\} = \{1.42, 1.81, 1.24, 0.21, 2.9\}$, which again we find by minimal tuning.

For the scalability of TTS for the fully-frustrated glass as shown in the upperleft subplot of Fig. 3 in the main text, we use an implementation that assigns $\square$ to be the checkered cubes, as an extension of the KBD algorithm in 2d. To ensure that we are not exploiting the tiling instances, we intentionally offset the checkered pattern for the long-term memory variables with the one used to generate the instances. Note that even if the checkered patterns for the tiling glasses and the long-term memory were the same, it would by no means represent an advantage. This is because the problem of finding a ground state for a tiling instance still appears to be NP-complete, even if the knowledge of the checkered pattern (or the planting pattern) is given. In fact, even for 2d glasses, there is not a single cluster algorithm that is efficient even when the checkered pattern is known. It is interesting to note that the scalability of both the bond and plaquette memory dynamics show similar scaling powers, as shown in Fig. 6.

SM E: Edwards-Anderson model

The Edwards-Anderson model is the “prototypical” spin glass where the couplings $J$ are sampled uniformly from $\{-1, +1\}$. It is believed that the lower critical dimension of this model is $d_l = 2.5$, meaning that it exhibits a non-trivial phase transition in 3d (even though this has not yet been mathematically proven). Therefore, the 3d EA model serves as a perfect “natural” benchmark for simulation algorithms. Note that since the model does not assume a planted ground state, extracting the time-to-solution (TTS) measure (see Section F 1) is infeasible (because it requires actually solving for the ground state which may be exponentially hard to verify). Therefore, we opt to simply monitor the evolution of the lowest Ising energy found by the algorithm so far as a function of time (see Section C and D for the measurement of simulation time) at a fixed size of $20 \times 20 \times 20$. 

Even though the EA model is not planted, the EA model is translationally invariant (or rather the disorder measure is), the system is self-averaging\textsuperscript{103}, meaning that the ground state energy in the thermodynamic limit equals the mean ground state energies over the distribution of disorder, or
\[
\lim_{N \to \infty} E_{gs}(\mathbf{J}) = \overline{E_{gs}(\mathbf{J})},
\]
where $E_{gs} (\mathbf{J})$ is the ground state energy of a particular realization $\mathbf{J}$. This value is extracted to be $-1.7875(2)$ (from simulations performed on 3d EA models sized from $L = 6$ to $L = 14$) normalized with the number of spins\textsuperscript{61}, so we can monitor the best Ising energies found by an algorithm so far over a sample of disorder realizations with the following metric
\[
\log(\delta u(t)) = \log \left( 1.7875 + \frac{E_{best}(\mathbf{J},t)}{N} \right),
\]
where $E_{best}$ is the best energy found so far in time $t$ for the realization $\mathbf{J}$. The overline in this context denotes both the average over the glass samples and the expected operation of the algorithm (if it is stochastic). Note that this metric should decrease monotonously in time, with a smaller value meaning that the algorithm is closer to the ground state energy. For our study, we randomly generated 400 EA samples of size $L = 20$.

It is important to note that this measure may become unreliable for measuring the absolute efficiency of an algorithm at low temperature, for small sample and lattice sizes, in which case the statistics of the glass sample itself (plus finite-size effects) becomes prominent. For instance, when an algorithm plateaus at some energy above the expected ground state, it is unclear whether this deviation may actually be due to the inability of the algorithm to find the ground state rather than the sample itself having a higher ground state energy than expected. And in rare cases where the ground state energy of the sample is lower than expected, the measure $\log(\delta u(t))$ then becomes undefined. Nevertheless, for sufficiently large sample and lattice sizes, it is still expected that the inefficiency of the algorithms will dominate the statistical deviations of the sample, making the logarithmic measure accurate in comparing the efficiency of the algorithms.

**SM F: Planted Ising Spin-glasses**

Planted Ising spin-glasses is a class of Ising instances generated by assuming a specific ground state, without sacrificing the glassy property of having a highly non-convex energy landscapes. This makes them ideal benchmarks for evaluating algorithms that simulate critical spin-glass dynamics, or finding the ground state of glass realizations. In addition, it is generally beneficial to have control over the hardness of the planted instances to offer different degrees of evaluation, and an important way of realizing this control is to design the class to offer tunable frustration ratio\textsuperscript{42,60,104}.

We would also like to address a common belief that the so-called “planted” glass models are not “real” spin glasses. The identification of what constitutes a “real spin glass” is meaningless, as formally speaking, any spin-glass structure is essentially a distribution of couplings $\mathbf{J}$ on some underlying graph structure\textsuperscript{105}, and practically speaking, there is no reason to expect planted models to be less “physical” than traditional glass models. In fact, multiple “planted” structures are known to exhibit glass-like behaviors\textsuperscript{1,106}, and even some highly frustrated deterministic models are glassy in nature\textsuperscript{107}. In another work, we show several interesting extremal properties of planted glasses in 2d\textsuperscript{28}.

A full discussion of the computational hardness of planted problems is beyond the scope of this work, but we refer the reader to\textsuperscript{100,108} for a formal analysis on the computational hardness of planted problems. Generally speaking, it is unknown whether we can have a method of randomly generating problem instances in the NP-complete class (planted or not) that are self-reducible\textsuperscript{109}, meaning that it is unclear we can generate hard\textsuperscript{9} instances with the average complexity equal to the worst-case complexity. Nevertheless, empirical evidence has suggested that the fully-frustrated glass is extremely difficult for most state-of-the-art algorithms to simulate or solve\textsuperscript{42,101}.

\textsuperscript{9} “Hard” in the sense that the problem is NP-complete. If an NP-complete problem is non-adaptively random self-reducible, then the polynomial hierarchy collapses to $\Sigma_3$. The problem of finding the ground state of planted instances is clearly not NP-complete, so it is possible that they are self-reducible (without any severe implication).
1. Time-to-solution

Generally speaking, there are two major ways to evaluate the efficiency of an optimizer/solver in its ability to discover a ground state of a glassy instance. The first way is to allocate the solver a certain amount of time, and allow the solver to run until timeout. This evaluation method is generally done for incomplete solvers \cite{10}, where the goal is to test the capability of the solver to reach the lowest energy possible within a given time. In most cases, this method of evaluation does not provide an accurate measure of the efficiency of the solver or the complexity of the problem class, and is generally biased towards greedy or local solvers. To see why, most complex systems (such as the Ising spin glass) admit a rough energy landscape with an abundance of local minima (metastable states) which can be readily accessed by a greedy algorithm with random initial conditions convoluted by random noise. However, the transition from a metastable state to the global optimum (Ising ground state) requires exponential cost in time \cite{09}, and, in addition, requires careful coordinated non-local updates \cite{24,26,31,38}.

A greedy solver may reach a metastable state relatively quickly \cite{111}, but it may never reach the global optimum. On the other hand, a solver with collective dynamics may sacrifice some time to carefully establish long-range connections \cite{38}, and eventually reach the global optimum after being allocated sufficient time. Therefore, a more faithful measurement of the efficiency of a solver is to record the time it takes for the solver to return the global optimum (or reach a certain gap above the optimum). This evaluation is commonly known as the time-to-solution (TTS) evaluation \cite{104}. However, to actually perform this measurement in practice, one has to know in advance what the optimum is. A way to achieve this goal is to assume (or plant) a solution in advance, and generate instances such that the optimum can be easily extracted by the generator but exponentially hard for the solver to find \cite{109}. These planted instances can then be used to evaluate the performance of the solver, and also check the correctness of the solver by comparing its solution to the planted one.

2. The Tiling Glass

The tiling glass \cite{42} is a class of Ising spin glasses with a planted ground state energy. It is a class where the expected local residual entropy of the checkered cubes governs exponentially the hardness of the instances \cite{42,112}. This type of “planted” glass structure has rather rich equilibrium and non-equilibrium dynamics \cite{28,42,101}, making it ideal for analyzing the efficiency of spin-glass simulation algorithms. Getting back on track with the discussion on the tiling glass, we first note that the tiling glass allows for the generation of a “fully-frustrated” spin glass where every face of the 3d lattice is frustrated, meaning that one cannot simultaneously satisfy all interactions in any give face \cite{113}. More generally speaking, the frustration profile (or the sign) of 5 out of the 6 faces of a cube can be independently assigned. Therefore, one can also opt to frustrate only a portion of the 6 faces to decrease the frustration ratio (thus the expected hardness) of the planted instances. All the possible frustration profiles of a cube are enumerated in the original work detailing the tiling glass \cite{42}. In Fig. 3 of the main text, the “fully-frustrated” glass refers to the tiling glass assembled with $F_6$ cubes (where all 6 faces frustrated), and the “partially-frustrated” glass refers to the tiling glass assembled with $F_{24}$ cubes (where 4 out of 6 cubes are frustrated) \cite{42}.

Glossing over some caveats with the problem of defining full frustration in a 3d lattice \cite{28}, we note that a fully frustrated lattice under the tiling construction also attains the maximal local ground state degeneracy cube-wise, and this has been demonstrated numerically to generate an extremely rough energy landscape. It should be noted that the generation of a fully frustrated hypercube with extremal frustration is highly non-trivial in $d \geq 4$ dimensions \cite{114}, so we limit our studies to the 3d glasses. To assemble a fully-frustrated 3d glass from fully-frustrated cubes, the cubes can be rotated randomly and assembled in a checkered pattern to introduce further disorder. In general, the local frustration ratios of the cubes, fully- or partially-frustrated, is preserved when they are assembled to the global construction, meaning that the ground state energy of the glass is simply the sum of the local ground state energies of the cubes, which can be computed in linear time. Note that for state-of-the-art solvers \cite{18,31}, a fully frustrated construction already becomes computationally prohibitive to simulate in the periodic lattice sized $8 \times 8 \times 8$.

3. Effective Temperature Estimation

Even though the memory dynamics operate at non-equilibrium, in the sense that the effective temperature decreases monotonously in time (see Fig. 2 in the main text), the instantaneous distribution of internal energies over disorder and uniform initialization of the equations of motion does seem to converge to the Boltzmann distribution. This is reminiscent of the operation of simulated annealing (SA) \cite{16}, though there are major differences, particularly
FIG. 7: The estimated curves for the offseted intensive internal energy $u(T)$ as given in Eq. (F3), for fully frustrated $3d$ Ising glasses sized from $N = 4^3$ to $N = 6^3$. The curves slightly separate below the critical temperature $T_c \approx 1.2$ (as zoomed in on the lower right) before merging again in the low temperature phase, possibly due to non-averaging finite size effects induced by metastability in the glass phase (see Fig. 9). The simulation is performed over 400 disorder realizations with PT for $2^{21}$ sweeps and replicas spaced geometrically in temperature.

FIG. 8: The change of internal energy in time when the fully frustrated $3d$ Ising glass is evolved under memory dynamics. Statistics are gathered over 400 disorder realizations. The solid line represents the mean of the energy samples $\bar{u}$, and the shaded region represents the standard deviation of the samples $\hat{\sigma}_u$. Of particular importance is the observation that the energy variation decreases as the system size is increased, which is in line with the self-averaging property of the equilibrium internal energy, as expressed in Eq. (F1). Informally, this means that the energy variation is in fact a thermal property, instead of being induced by the random initialization of memory dynamics, which implies that the effective temperature curve shown in the left panel of Fig. 2 in the main text is in fact thermally robust.

in that SA relies on a carefully tuned annealing process approaching $T_c^{SA}$, while the transient memory dynamics dive below $T_c$ right away without sacrificing long-range order.

To see this ‘static’ equilibrium property more clearly, in Fig. 8 we observe that, for the memory dynamics, the relative variation of internal energies over disorder decreases as the system size is increased, which is an expected
FIG. 9: The curves show the Binder’s cumulant $g(T)$ for spin overlap $q$ with respect to temperature (see Eq. (F3)), for fully frustrated (FF) $3d$ Ising glasses sized from $N = 4^3$ to $N = 6^3$. The curves intersect at $T_c \approx 1.2$, which we shall interpret as the critical temperature of the $3d$ FF glass. Interestingly, $g(T)$ at the intersection point appears to be near 0, seeming to imply that replica symmetry is somehow “restored”, and the glass is fully disordered. This is a very interesting and uncommon phenomenon, as generally $g(T)$ is positive at the intersection point\(^{105}\). Furthermore, we see that $g(T)$ displays a pronounced dip below $T_c$, at varying values of $T$ depending on the system size, in a range roughly corresponding to the separation of energies as seen in Fig. 7. Again, this is possibly due to non-averaging finite size effects induced by metastability in the glass phase. We will make no attempt to investigate this phenomenon\(^{115}\), and simply show the empirical overlap distribution $P(q)$ at the low temperature phase, noting that this distribution becomes increasingly erratic as the temperature is lowered and the system size becomes smaller. The simulation is performed over 400 glass instances with ICM for $2^{21}$ sweeps with the replica pairs spaced geometrically in temperature.

property of an equilibrated spin glass as the internal energy is proven to be self-averaging\(^{116}\), or

$$\lim_{N \to \infty} \frac{U_J(T)}{U(T)} \to 1,$$

meaning that we can extract the effective temperature of the memory dynamics at any point of time from the sample mean of the recorded energies. This follows from the general procedure of maximum likelihood estimation (MLE)\(^{76,117}\),
where the following likelihood function is to be optimized over $\beta$,

$$
\sum_{\sigma \in S} P(\sigma) \log \left( \frac{e^{-\beta E(\sigma)}}{Z(\beta)} \right),
$$

where $S$ is the set of spin state samples. Note that we are ignoring the disorder distribution here for the aforementioned reasons. Setting the derivative of (F2) to zero gives us

$$
\frac{\partial}{\partial \beta} \left( \sum_{\sigma \in S} P(\sigma) \log \left( \frac{-\beta E(\sigma)}{Z(\beta)} \right) \right) = 0,
$$

where $\bar{E}$ is the sample mean of the energies. The temperature estimator is then given by

$$
\hat{T} = U^{-1}(\bar{E}),
$$

where again we are making no distinction between $U_J$ and $\bar{U}$. Note that $\hat{T}$ and $\hat{\beta}$ can be interchanged by virtue of the functional invariance of MLE estimators, and furthermore, the estimator $\hat{T}$ is efficient.

For non-trivial spin glasses, it is likely that the analytic form of $U(T)$ is not available, so we resort to using Monte Carlo methods to estimate it. For the tiling cubes, we can define the intensive internal energy offset by the planted ground state energy $E_0$ as

$$
u(T) = \frac{U(T) - E_0}{N},
$$

where $N$ is the number of spins.

### 4. Critical Temperature Estimation

For deterministic models (or sufficiently structured glasses), one can generally look at the behavior of the spin correlation or the aging profile of the overlap autocorrelation time to determine the critical temperature $T_c$. However, the local rotations of the tiling construction gives rise to large variations in these order parameters over different instance realizations (even in the absence of any gauge transformation). Therefore, we will resort to using the Binder’s cumulant as a much more robust order parameter, which is related to the kurtosis of the overlap distribution over both the Boltzmann measure and the disorder of $J$,

$$
g = \frac{1}{2} \left( 3 - \frac{\langle q^4 \rangle}{\langle q^2 \rangle^2} \right),
$$

where $\langle \cdot \rangle$ denotes the Boltzmann average, the overline denotes the quenched average over disorder, and $q$ is the spin overlap defined as

$$
q = \frac{1}{N} \sum_i \sigma_i^\alpha \sigma_i^\beta,
$$

with $\alpha$ and $\beta$ denoting two independent replicas. In most cases, the temperature at which the $g(T)$ curves intersect for different system sizes can be taken numerically as the critical temperature $T_c$ (see Fig. 9).

There may be a disagreement on whether a crossing at the cumulant of 0 actually accurately pinpoints the critical temperature of the fully-frustrated glass. This discussion is an open and important one, but far beyond the scope of this work. We will attempt to offer our side of this discussion in another work. Note that however, the exact value of the critical temperature $T_c$ is not central to the results here, which is to show the non-equilibrium critical property of memory, and prove that it can be leveraged to explore the low-temperature phase of spin glasses efficiently. In the context of this work, the critical temperature is only important for identifying the “transient” stage of memory evolution (see left subplot of Fig. 2 in main text), which may be slightly prolonged if $T_c$ is in fact lower.