Capturing exponential variance using polynomial resources: applying tensor networks to non-equilibrium stochastic processes

T. H. Johnson, T. J. Elliott, S. R. Clark, and D. Jaksch

1 Centre for Quantum Technologies, National University of Singapore, 3 Science Drive 2, 117543, Singapore
2 Clarendon Laboratory, University of Oxford, Parks Road, Oxford OX1 3PU, United Kingdom
3 Keble College, University of Oxford, Parks Road, Oxford OX1 3PG, United Kingdom
4 Institute for Scientific Interchange, Via Alassio 11/c, 10126 Torino, Italy

(Dated: April 7, 2015)

Estimating the expected value of an observable appearing in a non-equilibrium stochastic process usually involves sampling. If the observable’s variance is high, many samples are required. In contrast, we show that performing the same task without sampling, using tensor network compression, efficiently captures high variances in systems of various geometries and dimensions. We provide examples for which matching the accuracy of our efficient method would require a sample size scaling exponentially with system size. In particular, the high-variance observable $e^{-\beta W}$, motivated by Jarzynski’s equality, with $W$ the work done quenching from equilibrium at inverse temperature $\beta$, is exactly and efficiently captured by tensor networks.

Introduction.—Dynamical stochastic processes are used throughout the natural and social sciences when inaccessible degrees of freedom are well-represented by random variables $\{1, 2\}$. To calculate expected observable values, numerical methods are usually required. Out of equilibrium, the typical method is dynamical Monte Carlo $\{3, 4\}$. Essentially, averaging over randomly sampled paths provides an unbiased estimate for the expected value of an observable. To obtain a fixed expected fractional error, the number of paths sampled must scale linearly with the variance divided by the square of the expected value. For a multitude of important observables, such as those appearing in the estimation of free energies $\{5\}$ and likelihoods of rare events $\{11, 12\}$, this ratio is large: such observables are said to have high variance and sampling methods struggle when applied to them.

Here we present an approach that is very different to sampling. We simultaneously follow all paths, which is made efficient by using controlled data compression, usually approximate but exact in special cases, based on tensor networks. While tensor networks have previously been used in conjunction with stochastic processes $\{10–20\}$, the question of how their performance relates to variance has remained unanswered. Understanding this is crucial if we are to know whether or not tensor networks, which have had a revolutionary effect in simulating quantum systems $\{21, 27\}$ and have been used to great effect in solving partial differential equations $\{28, 31\}$, provide a useful and perhaps essential complementary technique to sampling in stochastic processes.

In this Letter we address this question and our answer is very clear: high variance does not limit the accuracy of tensor network compression, and tensor networks can be applied efficiently to tackle problems, even out of equilibrium, for which sampling-based methods struggle. This opens the door for the use of tensor network methods on a wide-variety of non-equilibrium stochastic systems for which capturing high variance is essential. In particular, we show that a distribution of weighted expectation values of high-variance observable $e^{-\beta W}$, with $W$ the work done quenching from equilibrium at inverse temperature $\beta$, is represented exactly by a highly compressed tensor network.

We focus on an Ising system, an example of which is shown in Fig. 1(a). It comprises $N$ nodes, labeled by $\ell$, the configuration $z_\ell$ of each taking one of $d = 2$ discrete values $z_\ell = \{-1, 1\}$. The configuration of all $N$ nodes is given by the $N$-tuple $z = (z_1, \ldots, z_N)$ and the probability of being in configuration $z$ is $P(z)$. Tensor networks best suit systems for which crucial quantities, like energy, are $n$-bodied, with $n$ small. We consider the simplest non-trivial case of an energy comprising single and two-body terms

$$E(z) = -J \sum_{(\ell, \ell') \in \mathcal{E}} z_\ell z_{\ell'} - \lambda \sum_\ell z_\ell,$$

where $\mathcal{E}$ are $N_\mathcal{E}$ edges connecting interacting nodes.

FIG. 1. (color online) Tensor network compression. (a) An Ising system whose degrees of freedom (blue circles with arrows) interact, in this case, with a two-dimensional lattice geometry (red lines). (b) The probability distributions $P_E(z)$ and $P(z, t)$, and $Q(z, t)$ (see main text) at any time $t$ are compressed by representing them (approximately or, in special cases, exactly) by a contraction of tensors (green circles) with the same geometry as the interactions. Each black leg corresponds to an index of a tensor, and the joining of two legs represents the contraction of the two corresponding indices.
Equilibrium. — The relationship between compressibility and variance out of equilibrium builds on that in equilibrium. The equilibrium Gibbs distribution at inverse temperature $\beta$ is $P_E(z) = e^{-\beta E(z)}$, normalized to the partition function $Z_E = \sum_z e^{-\beta E(z)}$. It is always possible to represent $P_E(z)$ (or any distribution) by a tensor network of the form shown in Fig. 1(b)

$$P_E(z) = \sum_i \prod_\ell A_i^{[\ell]} z_\ell,$$

that shares the same geometry as the interactions. Here $A^{[\ell]}_i$ is a tensor associated with node $\ell$. It has a physical index $z_\ell$ and $k_\ell$ auxiliary indices $i_\ell$, one for each edge connected to $\ell$, and each taking one of $\chi$ values, which may in principle be large. The sum is over the values taken by all auxiliary indices, which is just a sum over an $N_\ell$-tuple of indices. The Gibbs distribution is important because the tensor network representation is exact for $\chi = d$ (see Supplemental Material [32]). This implies that $\sum \delta^{d_N+1}$ numbers may be used to represent $d^N$ others, providing a significant yet exact compression if the degrees $k_\ell$ are limited, as in lattice systems with local interactions.

As well as compression, tensor networks offer a means of calculating the partition function $Z_E$, since $Z_E = \sum_z P_E(z) = \sum_i \prod_\ell T_i^{[\ell]}$ with transfer tensors $T_i^{[\ell]} = \sum_{z_\ell} A_i^{[\ell]} z_\ell$. For a one-dimensional (1D) chain the partition function $Z_E$ relates to a product of transfer matrices and requires $O(N d^2)$ or $O(N d^3)$ resources to compute for open or periodic boundaries, respectively. In higher dimensions (if the tensor network has a large treewidth [37, 38]) the tensor contractions cannot in general be performed both exactly and efficiently, but efficient strategies exist to perform them approximately. Levin and Nave [35] demonstrated that this can be done accurately for two-dimensional (2D) non-critical lattice systems using tensor renormalization group [33, 40].

Contrastingly, estimating the partition function $Z_E$ directly by evaluating the sum $Z_E = \sum_z P_E(z)$ through random sampling is made difficult by the fact that, in general, the variance of observables requiring estimation grows quickly with system size [2]. The tensor network representations of $P_E(z)$ and $Z_E$ show that high variance does not imply difficulty in equilibrium, away from criticality.

Non-equilibrium. — Out of equilibrium, the dynamics of a Markovian system [41] depends only on its current configuration, and the evolution of the distribution $P(z, t)$ is described by a master equation of the form

$$\frac{\partial P(z, t)}{\partial t} = \sum_{z'} H(z, z', t) P(z', t).$$

Each non-negative off-diagonal element $H(z, z', t)$ for $z \neq z'$ is the Poisson rate of a transition from $z'$ to $z$ at time $t$, and together these fix the non-positive diagonals $H(z, z, t) = -\sum_{z' \neq z} H(z', z, t)$ such that the normalization of $P(z, t)$ is conserved. $H$ is commonly referred to as the Hamiltonian.

To simulate such dynamics using non-equilibrium tensor network methods, we represent $P(z, t)$ at any time by a tensor network, as in Eq. (2), with a small $\chi$. Doing so assumes that this representation, while not necessarily exact, is accurate. There is no guarantee of this accurate compressibility on all occasions, but it is expected in many situations. For example, consider a quench from one Hamiltonian $H(z, z', 0) = H_0(z, z')$ to another $H(z, z', \tau) = H_1(z, z') \neq H_0(z, z')$, where the system begins in the stationary state $P_0(z)$ satisfying $\sum_z H_0(z, z') P_0(z') = 0$. For much later times $t \gg \tau$ (on the timescale $h^{-1}$, where $h$ is some Hamiltonian-specific convergence rate), the system will converge to another stationary state $P_1(z)$ satisfying $\sum_z H_1(z, z') P_1(z') = 0$.

Numerous examples have revealed that stationary states of local stochastic processes are accurately compressible via tensor network representations [10, 21]. Thus in such quenches both initial and long-time distributions $P_0(z)$ and $P_1(z)$ are accurately compressible. Unlike for quantum systems [42], compression errors are limited even when a system is driven away from equilibrium.

The probability distribution $P(z, t)$ over configurations contains only partial information about the full probability distribution over the possible paths through configuration space taken by the stochastic process. As such, the expected values of only certain observables may be calculated from $P(z, t)$. These include observables whose values $O(z)$ depend on the configuration $z$ of the system at a single time $t$, thus having expected value $\langle O(t) \rangle = \sum_z O(z) P(z, t)$. We call such observables configuration-dependent and use the example of the magnetization $M(z) = \sum_z z_i$. The values of some other observables depend on the full path taken by the system and their expected values cannot be calculated from $P(z, t)$. We call such observables path-dependent, and use the example of the work done $W(t) = -\int_0^t ds M(z(s)) \dot{z}(s)$ by varying $\lambda(t)$ between times 0 and $t$.

Although not previously considered in the literature, the expected values of some path-dependent observables can indeed be calculated using tensor networks and, as we will show, provide us with a stark example of exact compressibility in the face of high variance out of equilibrium. The idea is to represent the relevant path-dependent information locally in time, not with $P(z, t)$, but through the distribution of weighted conditional expected values $Q(z, t) = P(z, t) \langle O(z, t) \rangle$. Here $\langle O(t) \rangle = \sum_z Q(z, t)$ is then obtained from $Q(z, t)$. It follows from Eq. (3) that the distribution $Q(z, t)$ evolves
evolution using tensor networks. Consider the quench in
the same of the former. Additionally, \( \dot{\lambda} \) thus the accurate compressibility of the latter implies the
correlation timescale \( h \langle P \rangle \) for times \( z \leq h \tau \). (d) Similarly, the fractional error \( \epsilon_2 \) between calculating \( \langle e^{-2\beta W(t)} \rangle \) by summing \( Q(z, t) \) and its
compressed tensor network approximation. The parameters
used are \( \beta \lambda_0 = 0, \beta \lambda_1 = 1, h\tau = 10, \beta J = 1 \) and \( N = 8 \).

as

\[
\frac{\partial Q(z, t)}{\partial t} = \sum_{z'} H'(z, z', t)Q(z', t),
\]

with \( H'(z, z', t) = H(z, z', t) + \dot{\lambda}(z, t)\delta(z, z') \), where \( \dot{\lambda}(z, t) \)
is the rate of increase of the natural logarithm of the
observable at configuration \( z \) and time \( t \).

It is desirable to predict, as we have for \( P(z, t) \), the
accuracy of compressing \( Q(z, t) \) at any time during its
evolution using tensor networks. Consider the quench in
\( \lambda(t) \) between times \( 0 \) and \( \tau \), starting from equilibrium.
The distributions are initially equal \( Q(z, 0) = P(z, 0) \),
thus the accurate compressibility of the latter implies the
same of the former. Additionally, \( \dot{\lambda}(z, t) \) is only non-zero
for times \( t < \tau \) and the stochastic evolution is ergodic.
Thus after a sufficiently long time \( t \gg \tau \) (relative again to
convergence timescale \( h^{-1} \)) the configurations will have
mixed such that \( \langle O(z, t) \rangle = \langle O(t) \rangle \) is independent of \( z \)
and thus once again \( Q(z, t) = \langle O(t) \rangle P(z, t) \) is as
accurately compressible as \( P(z, t) \).

Numerical examples.—We demonstrate these behaviors
for a system undergoing thermalizing Glauber dynamics \( 43 \) via local transitions,

\[
H_E(z, z') = h(z, z') \left[ 1 + e^{-\beta(E(z') - E(z))} \right]^{-1},
\]

for \( z \neq z' \). Here \( h(z, z') = h(z', z) \) are symmetric rates
equaling a non-zero rate \( h \) only where \( z \) and \( z' \) differ
by the configuration of a single node. The energy is
quenched via the parameter \( \lambda(t) \) varying from \( \lambda_0 \) to \( \lambda_1 \)
over time \( 0 \leq t \leq \tau \) according to a smoothed tanh ramp
(see Supplemental Material \( 32 \)), as drawn in Fig. 2(b). We focus on configuration-dependent observable \( e^{M(t)} \)
and path-dependent observables \( e^{-2\beta W(t)} \) and \( e^{-2\beta W(t)} \).

All have variance over mean squared growing exponential
with system size \( N \). Initially, we consider the Ising nodes to be
in an open 1D chain, illustrated in Fig. 2(a).

To assess the accuracy of compression, we exactly cal-
culate distributions \( P(z, t) \) and \( Q(z, t) \) for small \( N = 8 \) at
time \( t \), then calculate the error in the expected value of observables induced by compressing the distributions as
a tensor network. The errors shown in Figs. 2(c) and (d)
for \( e^{M(t)} \) and \( e^{-3\beta W(t)} \), respectively, show that, despite large variances, expected values are relatively unaffected
by tensor network compression. The distributions are ex-
actly compressible at \( t = 0 \) and thus no error occurs, as
expected. The errors due compression initially increase as \( \lambda(t) \) varies, then decrease exponentially to small values
again on a timescale \( \sim h^{-1} \). Interestingly, errors begin to
decrease even at times \( t < \tau \) prior to the end of the quench. For \( \chi \geq 4 \) the compression is near-exact at all
times. We arrive at similar conclusions for other types of
variation tried e.g. linear, variations in \( J \), and other
observables.

A striking example is found in the path-dependent observable \( e^{-\beta W(t)} \). The observable has received
particular attention due to its featuring in several non-
equilibrium identities in statistical physics, such as that
by Jarzynski \( 44 \). Crucially for our discussion, the
non-equilibrium distribution \( Q(z, t) = P(z, t)(O(z, t)) \) for
this special case has an equilibrium structure \( Q(z, t) = P(z)Z_i(\beta)/Z_0(\beta) \),
where we have used shorthand of the form \( P(z) \) for the Gibbs distribution corresponding to \( \lambda(t) \) and \( Z_i(\beta) \) for the corresponding partition function
(where from now on we normalize the Gibbs distribution to 1). It immediately follows from our discussion of systems in equilibrium that \( Q(z, t) \), despite containing information about non-equilibrium high-variance observables, has an exact highly-compressed \( \chi = d \) tensor
network representation at all times \( t \). This can be used to
efficiently and accurately calculate not only \( \langle O(t) \rangle \) but
a range of properties of the work distribution during such
dynamics. Note that this exact behavior is particular to
\( e^{-2\beta W(t)} \) and doesn’t even extend to its square \( e^{-3\beta W(t)} \).

We have so far demonstrated accurate single-time compressibility. We next examine how this extends to a
dynamical tensor network simulation, where compression of \( P(z, t) \) or \( Q(z, t) \) occurs not only at a single time but
at all times during their evolution. While one might expect
the compression errors at single times to accumulate,
we find this is mitigated by the ergodicity of the
evolution (unlike in quantum systems). For example, er-
rors in \( P(z, t) \) will not change the distribution to which it
converges, and thus the significance of transient er-
rors diminish, rather than accumulate, in time. In what
follows, the specific algorithm we use to perform the evolu-
tions of Eqs. \( 3 \) and \( 4 \) is time-evolution block deci-
mation (TEBD) \( 20, 45, 46 \). The TEBD algorithm uses a
timestep \( \delta t \) resulting in an error, beyond that due to
compression, of \( O(N\delta t^2) \) and requires time \( O(N\chi^3\delta t^{-1}) \) (see
the Supplemental Material \( 32 \)).

We first calculated \( \langle e^{M(t)} \rangle \) and \( \langle e^{-2\beta W(t)} \rangle \), where \( M = \)
so matching our cpu time would require each path to be

\[ \epsilon_3 \approx \frac{1}{N} \]

We do not compare this against the cpu time of any one
takes less than an hour and achieves an error
fractional error \( \epsilon \)

Compression and variance
ative to the
\[ \chi \]

\[ \langle \chi \rangle \]

\[ \tau \]

\[ \beta W \]

the same fractional accuracy (namely, \( \epsilon = 10 \))

\[ \langle \chi \rangle \]

\[ \langle \chi \rangle \]

\[ (\beta W t) \]

\[ \delta t \]

\[ (\beta W t) \]

\[ \delta t \]

\[ h \]

\[ \beta W \]

\[ \beta W \]

\[ \beta \lambda \]

\[ h \tau \]

\[ \beta \]

\[ \beta J \]

\[ N = 200 \]

\[ h \delta t = 10^{-3} \]

\[ M(\tau) \]

\[ W = M(\tau) \]

\[ \chi \]

\[ \|

\[ S \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]

\[ \chi \]
sical stochastic systems are currently in an even better position than quantum systems to benefit from current high-dimensional tensor network algorithms.

THJ, SRC, and DJ thank the National Research Foundation and the Ministry of Education of Singapore for support. TJE thanks St Anne’s College of the University of Oxford for financial support. This research received funding from the ERC under the European Unions Seventh Framework Programme (FP7/2007-2013)/ERC Grant Agreement no. 319286 Q-MAC and from the EPSRC through projects EP/K038311/1 and EP/J010529/1.

SUPPLEMENTAL MATERIAL

1. Exact tensor network for a Gibbs distribution

It is possible to represent the Gibbs distribution exactly by a tensor network

\[ P_E(z) = \sum_i \prod_\ell A[i]^{[\ell]z_\ell}, \]

where the bond dimensions are \( \chi = d \).

The generality of such a representation follows from the arguments justifying the generality of the tensor network factorization of the partition function as, for example, appears in Ref. [32]. Here we give explicit expressions for the tensors \( A[\ell] \) corresponding to the two systems discussed in the main text: the one-dimensional (1D) open Ising chain and the two-dimensional (2D) periodic square Ising system, both with \( d = 2 \).

For the 1D open Ising chain the elements of tensors \( A[\ell] \) are

\[
\begin{align*}
A_{i_1}^{[1]z_1} &= \exp[\beta z_1(J_{i_1}/2 + \lambda)], \\
A_{i_{\ell-1}i_\ell}^{[\ell]z_\ell} &= \exp[\beta z_\ell(J(i_{\ell-1} + i_\ell)/2 + \lambda)], \quad 1 < \ell < N, \\
A_{i_{N-1}}^{[N]z_N} &= \exp[\beta z_N(Ji_{N-1}/2 + \lambda)].
\end{align*}
\]

The labeling of the indices \( i_\ell, \) for \( \ell = 1, \ldots, N - 1 \), makes it clear which pairs are contracted. The indices take values \( \pm 1 \) and so the matrices \( A[\ell]^{z_\ell} \) for \( 1 < \ell < N \) are \( 2 \times 2 \) (the boundary vectors \( A[1]^{[1]z_1} \) and \( A[N]^{[N]z_N} \) are \( 1 \times 2 \) and \( 2 \times 1 \), respectively).

For the 2D periodic square Ising system the tensors \( A[\ell] \) are identical and their elements are given by

\[
A_{i_1i_2i_3i_4}^{[\ell]z_\ell} = \exp \left[ \beta z_\ell \left( J \sum_{\ell' = 1}^{4} i_{\ell'}/2 + \lambda \right) \right].
\]

The indices should be paired to form a square periodic lattice. The order of the indices at any one site is not important since the tensors are symmetric in the indices \( i_1, \ldots, i_4 \). As before, the indices \( i_{\ell'} \) for \( \ell' = 1, \ldots, 4 \) take values \( \pm 1 \).

2. Time-dependence of the bias

In our calculations the bias \( \lambda(t) \) is varied between \( \lambda_0 \) and \( \lambda_1 \) according to a smoothed tanh ramp. The exact time-dependence used is

\[
\lambda(t) = \lambda_0 + \frac{\lambda_1 - \lambda_0}{2} \left[ 1 + \frac{\tanh \left( \left( \frac{t}{\tau} - \frac{1}{2} \right) \pi \right)}{\tanh(1)} \right],
\]

where \( \tau \) is the duration of the quench.

3. Single-time compression

With the system out of equilibrium, we wish to determine the effect of compressing a distribution \( P(z, t) \) (similarly \( Q(z, t) \)) at some time \( t \). In other words we wish to construct a tensor network of small dimension \( \chi \) that closely approximates the distribution

\[ P(z, t) \approx \sum_i \prod_\ell A[i]^{[\ell]z_\ell}. \]

For a 1D system with open boundaries the tensor network takes the matrix product form

\[
\sum_i \prod_\ell A[i]^{[\ell]z_\ell} = A[[1]z_1]A[[2]z_2] \ldots A[[N]z_N],
\]

with \( \chi \times \chi \) matrices \( A[\ell]^{[\ell]z_\ell} \) for \( 1 < \ell < N \), and \( 1 \times \chi \) and \( \chi \times 1 \) boundary vectors \( A[[1]z_1] \) and \( A[[N]z_N] \). It is well-known how to find an accurate matrix product approximation to a distribution \( P(z, t) \) using singular value decomposition decimation, as explained in Refs. [20, 45]. Note that the matrix product obtained may not be optimally accurate for the observable of interest, and thus even more effective compression may be possible than that we have reported. However, singular value decomposition decimation is closely related to how compression occurs in the operation of the time-evolution block decimation (TEBD) algorithm, and thus we expect the conclusions regarding compressibility at a single time here to be most relevant to repeated compression within the TEBD algorithm.

4. Time-evolution block decimation

TEBD is a standard method, detailed in Refs. [20, 45], for evolving a matrix product representation of some distribution \( P(z, t) \) according to an equation of the form

\[
\frac{\partial P(z, t)}{\partial t} = \sum_{z'} H(z, z', t)P(z', t).
\]

The algorithm proceeds by breaking the evolution into timesteps \( \delta t \). Evolving the distribution \( P(z, t) \) over a timestep takes it away from one expressed in terms of
a matrix product of dimension \( \chi \). Thus the distribution is repeatedly, within each timestep, re-compressed to this form using the singular value decomposition. The method we use to discretize time is a typical one and is explained in detail in Ref. [21]. It is accurate up to errors that are second-order in \( \delta t \).

The most typical implementations of TEBD are compatible with two-site nearest-neighbor Hamiltonians only. Since the values of the transition rates appearing in Glauber dynamics depend on the energies at three nodes, the Hamiltonian is not naturally two-site. To bring it into standard form, rather than \( N \) single nodes with \( d = 2 \), the system is broken down into \( N/2 \) supernodes with \( d = 4 \) according to \( z = (z_{1,2}, \ldots, z_{N-1,N}) \), where \( z_{2\ell-1,2\ell} = \{(−1,−1), (−1,1), (1,−1), (1,1)\} \) for \( \ell = 1, \ldots, N/2 \). In the supernode picture the Hamiltonian is naturally formed of two-site nearest-neighbor terms.

Finally, the calculation results and scaling of resources quoted in the main text is for fixed IEEE 64-bit double precision computing. Though not necessary for the calculations presented here, higher than normal precision may be required for very large systems (see e.g. Ref. [17]) due to the number of orders of magnitude spanned by the observable and probability distributions.

5. Tensor renormalization group

We have used the tensor renormalization group method to calculate the sum \( \sum_{z \in S} P(z, t) \) for the case of an \( N = 64 \times 64 \) periodic square Ising system and \( S \) the set of configurations with \( z_1 = 1 \). We are able to use tensor renormalization group methods because of the equilibrium structure \( P(z, t) = P_t(z)Z_t(\beta)/Z_0(\beta) \) taken by the non-equilibrium distribution \( Q(z, t) \) for this observable.

To see this, we use the above to rewrite \( \sum_{z \in S} Q(z, t) = (Z_t(\beta)/Z_0(\beta)) \sum_{z \in S} P_t(z) \), where \( P_t(z) \) is the equilibrium Gibbs distribution corresponding to the bias \( \lambda(t) \) at time \( t \), and \( Z_t(\beta) \) the corresponding partition function. Thus the calculation reduces to calculating partition functions \( Z_t(\beta) \) and partial sums \( \sum_{z \in S} P_t(z) \) over Gibbs distributions conditioned upon the value of a single spin \( z_1 = 1 \). Given the representation of a Gibbs distribution \( P_t(z) \) by a tensor network, the partition function \( Z_t = \sum_{z} P_t(z) = \sum_{i} \prod_{l} T_{i}^{[l]} \) is merely a contraction of transfer tensors \( T^{[l]} = \sum_{z_i} A^{[l]z_i} \) with a square geometry. The partial sum \( \sum_{z \in S} P_t(z) = \sum_{i} \prod_{l} T_{i}^{[l]} \) is similar except with \( T^{[1]} = A^{[1]z_1} \).

These contractions can be performed using the well-known tensor renormalization group method [35], the relevant details about which we provide here. The strategy is first to perform the contractions involving each set of four nodes making up a node of a coarse-grained square lattice. We are then left with tensor network in a square lattice with the number of nodes reduced by a factor of four, but the dimension of each index potentially increased by a power of 4. Next, we compress this to obtain smaller index dimension \( \chi \). The contraction/compression process is then repeated iteratively until a single node remains, and the contraction can be trivially evaluated.

For \( N = 64 \times 64 \) this requires six iterations. We found it acceptable to use an intermediary dimension \( \chi = 3 \).

6. Dynamical Monte Carlo

We have used dynamical Monte Carlo to calculate the sum \( \sum_{z \in S} P(z, t) \) for the case of an \( N = 64 \times 64 \) periodic square Ising system and \( S \) the set of configurations with \( z_1 = 1 \). We were able to use dynamical Monte Carlo because, by defining an observable \( O(z) \) taking values 1 if \( z \in S \) and 0 otherwise, we have \( \sum_{z \in S} P(z, t) = \sum_{z} O(z)P(z, t) \) i.e. the quantity of interest is just the expected value \( \langle O(t) \rangle \) of this observable.

We implement a variable-timestep dynamical Monte Carlo algorithm (as e.g. explained in Ref. [3]) to sample \( N_s \) paths and estimate \( \langle O(t) \rangle \) by averaging the values taken by the observable over such paths. Because the variance of the observable is necessarily less than unity, the expected error due to insufficient sampling must be less than \( 1/\sqrt{N_s} \), which is small for the \( N_s = 14790 \) we used.

The only source of error other than insufficient sampling is an approximation made when estimating the time until the next transition as part of the sampling of a path. We assumed for simplicity that the transition rates stayed constant between transitions, which ignores their varying due to the change in \( \lambda(t) \). Since the average time between transitions is on the order of \( (64^2 \hbar)^{-1} \) and \( \beta \lambda \) is varied by unity over a time \( 10 \hbar^{-1} \) in our example, the approximation induced by this is negligible.


[32] See Supplemental Material submitted alongside this manuscript.
[48] Longer range but still local interaction could also be treated using tensor networks.
[49] The values of $A[\ell]$ follow from similar arguments to those for the tensor network factorization of the partition function [33].
[50] The exact time dependence used is

\[ \lambda(t) = \lambda_0 + \frac{\lambda_1 - \lambda_0}{2} \left[ 1 + \frac{\tanh \left( \frac{\ell - t \Delta}{\lambda_1 - \lambda_0} \frac{\pi}{2} \right)}{\tanh (1)} \right]. \]

[51] This can be extended easily to include observables whose values change multiplicatively when a transition occurs.
[52] To allow us to use nearest-neighbor gates only, rather than $N$ single spins with $d = 2$, the system is broken down into $N/2$ superspins with $d = 4$ according to $z = (z_1, z_2, \ldots, z_{N-1}, N)$, where $z_{2\ell-1, 2\ell} = \{(-1, -1), (-1, 1), (1, -1), (1, 1)\}$ for $\ell = 1, \ldots, N/2$.
[53] The calculation results and scaling of resources quoted here is for fixed IEEE 64-bit double precision computing. Though not necessary for the calculations presented here, higher than normal precision may be required for very large systems due to the number of orders of magnitude spanned by the observable and probability distributions.