The Journal of Chemical Physics

Evolutionary reinforcement learning of dynamical large deviations

Cite as: J. Chem. Phys. **153**, 044113 (2020); https://doi.org/10.1063/5.0015301
Submitted: 27 May 2020 . Accepted: 31 May 2020 . Published Online: 27 July 2020

Stephen Whitelam, Daniel Jacobson, and Isaac Tamblyn

The Journal of Chemical Physics ARTICLE scitation.org/journal/jcp

Evolutionary reinforcement learning of dynamical large deviations

Cite as: J. Chem. Phys. 153, 044113 (2020); doi: 10.1063/5.0015301
Submitted: 27 May 2020 • Accepted: 31 May 2020 •
Published Online: 27 July 2020

Stephen Whitelam,[1,a)] Daniel Jacobson,[2] and Isaac Tamblyn[3,4]

AFFILIATIONS

[1]Molecular Foundry, Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley, California 94720, USA
[2]Division of Chemistry and Chemical Engineering, California Institute of Technology, Pasadena, California 91125, USA
[3]National Research Council of Canada, Ottawa, Ontario K1N 5A2, Canada
[4]Vector Institute for Artificial Intelligence, Toronto, Ontario M5G 1M1, Canada

[a)]**Author to whom correspondence should be addressed:** swhitelam@lbl.gov

ABSTRACT

We show how to bound and calculate the likelihood of dynamical large deviations using evolutionary reinforcement learning. An agent, a stochastic model, propagates a continuous-time Monte Carlo trajectory and receives a reward conditioned upon the values of certain path-extensive quantities. Evolution produces progressively fitter agents, potentially allowing the calculation of a piece of a large-deviation rate function for a particular model and path-extensive quantity. For models with small state spaces, the evolutionary process acts directly on rates, and for models with large state spaces, the process acts on the weights of a neural network that parameterizes the model's rates. This approach shows how path-extensive physics problems can be considered within a framework widely used in machine learning.

I. INTRODUCTION

Machine learning provides the physics community with methods that complement the traditional ones of physical insight and manipulation of equations. Many-parameter ansätze, sometimes encoded in the form of neural networks, can learn connections between physical properties (such as the positions of atoms and a system's internal energy) without drawing upon an underlying physical model.[1–15] Reinforcement learning is a branch of machine learning concerned with performing actions so as to maximize a numerical reward.[16] It has a close connection to ideas of stochastic control enacted by variational or adaptive algorithms.[17–25] A recent success of reinforcement learning is the playing of computer games.[26–43] Here, we show that reinforcement learning can also be used to propagate trajectories of a stochastic dynamics conditioned upon potentially rare values of a path-extensive observable. Doing so allows the calculation of dynamical large deviations, which are of fundamental importance, being to dynamical quantities what free energies are to static ones.[44–47]

Calculating large deviations is a challenging problem for which a variety of approaches have been developed.[17–21,23,24,44,45,48–53] Simplest and most general are the universal upper bounds on large-deviation rate functions for time-extensive observables.[50–52] Rate functions are proportional to the logarithmic probability with which rare fluctuations are realized, and the universal bounds are statements about how fluctuations of certain time-integrated quantities are constrained by the average values of other quantities. These universal bounds are not intended to be methods of accurate numerical sampling, and how closely they approximate the true rate function varies by system.[54] Numerical methods for calculating large deviations focus, by contrast, on the reconstruction of the scaled cumulant-generating function, the Legendre transform of the rate function. These methods include cloning,[48] cloning augmented by auxiliary dynamics,[23] adaptive methods,[24] and methods based on matrix product states.[53] Here, we work within the framework of the variational ansatz for rare dynamics (VARD) method of Ref. 54. In a technical sense, VARD can be viewed as a complement to other numerical methods in that it uses only auxiliary dynamics. From the standpoint of what it produces, it is in a sense intermediate between the methods for producing general bounds and those for numerically sampling rate functions. It provides a way of bounding the rate function by proposing and minimizing a variational

ansatz for the rare behavior of the problem under study. The better the ansatz, the tighter the bound, and if the ansatz is good enough, as determined by certain convergence criteria, the exact rate function can be recovered. We showed previously that simple, physically motivated ansätze containing 2 or 3 parameters, minimized by enumeration of parameter space, produce descriptive (i.e., tight) bounds on large-deviation rate functions for a selection of models, and from these bounds, the exact rate functions could be recovered.

Here, we show that evolutionary learning using multi-parameter ansätze, in some cases encoded by simple neural networks, provides tighter bounds still, which closely approximates the rate functions of a selection of models. Thus by moving between ansätze containing 1 or 2 parameters, and ansätze containing 10 or 20 parameters, we move between bounds of similar quality to the universal bounds[55] and bounds that are numerically close to the exact rate function. The latter bounds provide a good starting point for the calculation of the exact rate function (we provide two examples here) or the study of the associated rare dynamics. We focus here on the calculation of rate-function bounds using evolutionary learning, a procedure that is technically simple and does not require physical insight into the model under study. The alternative approach, in which we rely on physical insights for the construction of an ansatz, is described in Ref. 54.

II. LARGE DEVIATIONS BY CHANGE OF DYNAMICS

To set the large-deviation problem in a form amenable to reinforcement learning, consider a continuous-time Monte Carlo dynamics on a set of discrete states, with W_{xy} the rate for passing between states x and y, and $R_x = \sum_{y\neq x} W_{xy}$ the escape rate from x.[56] This dynamics generates a trajectory $\omega = x_0 \to x_1 \to \cdots \to x_{N(\omega)}$ consisting of $N(\omega)$ jumps $x_n \to x_{n+1}$ and associated jump times Δt_n. In the language of reinforcement learning, W_{xy} is a policy (often denoted by π) that stochastically selects a new state and a jump time given a current state.

Stochastic trajectories can be characterized by path-extensive observables $A = aT$, with

$$a = T^{-1}\sum_{n=0}^{N-1} \alpha_{x_n x_{n+1}}. \tag{1}$$

Here, α_{xy} is the change of the observable upon moving between x and y. This type of observable describes many physically important quantities, including work, entropy production, and non-decreasing counting observables.[45,57–60] Let the typical value of a be a_0, the limiting value of (1) for a long trajectory of the model W_{xy}. Finite-time fluctuations $a \neq a_0$ occur with a probability controlled by the distribution $\rho_T(A)$, taken over all trajectories of length T. For large T, this distribution often adopts the large-deviation form[44,46]

$$\rho_T(A) \approx \mathrm{e}^{-TJ(a)}. \tag{2}$$

$J(a)$ is the large-deviation rate function, which quantifies the likelihood of observing atypical values of a.[44,46] Calculation of $J(a)$ far from a_0 using only the original model is not feasible, because such values of a occur rarely. Instead, we can consider a new stochastic model, which we call the reference model, whose purpose is to allow the calculation of $J(a)$ potentially far from a_0.[21,61]

With the reference model, we can carry out a form of importance sampling.[17–20,44,46,54,61–68] Let the rates of the reference model be $\tilde{W}_{xy}$ and $\tilde{R}_x = \sum_{y\neq x} \tilde{W}_{xy}$, and let the limiting value of (1) for a long reference-model trajectory be $\tilde{a}_0$. Then, an upper bound on $J(a)$ at $a = \tilde{a}_0$ is given by the value of

$$J_0 = -T^{-1}\sum_{n=0}^{N-1} q_{x_n x_{n+1}} - T^{-1}\Delta q_N \tag{3}$$

for a long reference-model trajectory, where

$$q_{x_n x_{n+1}} = \ln\frac{W_{x_n x_{n+1}}}{\tilde{W}_{x_n x_{n+1}}} - \Delta\tilde{t}_n\left(R_{x_n} - \tilde{R}_{x_n}\right), \tag{4}$$

and $\Delta q_N \equiv -\left(T - \sum_{n=0}^{N-1}\Delta\tilde{t}_n\right)\left(R_{x_N} - \tilde{R}_{x_N}\right)$. Here, $\Delta\tilde{t}_n = -\ln\eta/\tilde{R}_{x_n}$ is the jump time of the reference model, and η is a random number uniformly distributed on (0, 1]. Equation (3) follows from straightforward algebra (see Appendix A). It can be motivated by noting that the probability of a jump $x \to y$ in time $\Delta\tilde{t}$ occurs in the reference model with probability (density) $\tilde{W}_{xy}\mathrm{e}^{-\tilde{R}_x\Delta\tilde{t}}$ and $W_{xy}\mathrm{e}^{-R_x\Delta\tilde{t}}$ in reference and original models; Eq. (3) is the sum over a trajectory of the log-ratio of such terms (plus a boundary term).

Our aim in this paper was to use evolutionary reinforcement learning to find a reference model (a new policy) $\tilde{W}_{xy}$ that produces a particular typical value of (1), say $\tilde{a}_0$, and which minimizes (3).

Given a value of $\tilde{a}_0$, the model associated with the smallest possible value of (3) is called the driven or effective model, and its typical behavior is equivalent to the conditioned rare behavior of the original model.[21] The typical behavior of the driven model yields, from (3), the piece $J(\tilde{a}_0)$ of the rate function of the *original* model at the point $a = \tilde{a}_0$. In previous work,[54] we showed that the reference model need only be *close* to the driven model (in a sense made precise in that paper) in order to calculate $J(\tilde{a}_0)$; from the typical behavior of such a reference model, we get a bound [Eq. (3)] $J_0(\tilde{a}_0) > J(\tilde{a}_0)$, and by sampling the atypical behavior of the reference model, we can (under certain conditions) compute a correction $J_1(\tilde{a}_0)$ such that $J_0(\tilde{a}_0) + J_1(\tilde{a}_0) = J(\tilde{a}_0)$. There, we showed that simple, physically motivated choices of reference model lead to relatively tight bounds ($J_0 \approx J$) and small corrections $J_1 \ll J_0$ for a set of models taken from the literature. In this paper, we show how to further improve the quality of these bounds using multi-parameter ansätze determined by evolutionary learning. We do not focus here on the calculation of the correction term (for more detail of that calculation, including convergence criteria, see Ref. 54), but we show for two of our examples that the correction term is indeed small. Error bars associated with the bound scale as $1/\sqrt{N}$, where N is the number of events in the trajectory. For the models studied here, we chose N large enough that error bars are smaller than symbol sizes.

The formulation of this section describes an extreme example of reinforcement learning in which there is no instantaneous reward, only an overall reward (or return) associated with the entire trajectory.[16] Given that we possess a constraint on a, work in continuous time, and the return depends explicitly on the policy, this problem also falls outside the (standard) Markov decision process framework.[69,70] A natural approach to such problems is evolutionary algorithms, which are simple to apply and have been shown to be competitive with gradient-based methods[16] on complex problems

whose solution requires upwards of thousands of parameters.[38,71–77] We use an evolutionary approach in this paper.

III. LARGE DEVIATIONS VIA EVOLUTIONARY REINFORCEMENT LEARNING

As proof of principle, we consider the example of entropy production in the 4-state model of Ref. 51. The model's rates do not satisfy detailed balance, and so it produces nonzero entropy on average.[78] The dynamical observable a is (1) with $\alpha_{xy} = \ln(p_{xy}/p_{yx})$, where $p_{xy} = W_{xy}/R_x$. In Fig. 1(a), we depict the model (the middle picture), with states x numbered clockwise from 1 at the top left. Red and blue links denote connections $x \to y$ with negative and positive entropy production, respectively, and the thickness of the links is proportional to the rate associated with the connection. The model's state space is small enough that the master operator can be solved by diagonalization,[44] yielding the exact rate function $J(a)$, shown as a black dashed line in Fig. 1(b).

We can reconstruct this function using evolutionary reinforcement learning by mutating the rates $\tilde{W}_{xy}$ of a set of reference models until certain values of (1) and (3) are achieved. The process is as follows:

We start by running a trajectory of the reference model (of $N = 10^4$ events) and recording the typical value of the observable and

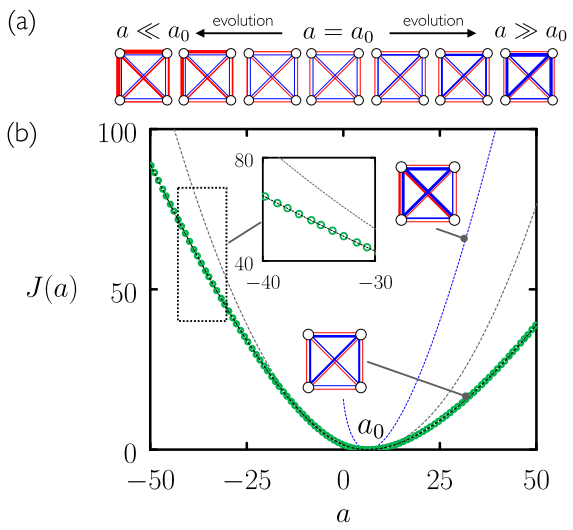

FIG. 1. (a) Evolutionary reinforcement learning can produce versions of the 4-state model of Ref. 51 whose typical dynamics are exactly equivalent to the rare dynamics of the original model (center) conditioned on values of entropy production a. (b) From these, we can calculate the corresponding large-deviation rate function, $J(a)$. The black dashed line is the exact answer, obtained by matrix diagonalization,[44,46] and the blue and gray dashed lines are the Conway–Maxwell–Poisson (CMP) bound[52] and the universal current bound,[50,51] respectively. The green points describe a bound resulting from a set of models generated by evolutionary reinforcement learning; this bound is effectively exact. Each green point is calculated using a single trajectory of a stochastic model produced by the evolutionary process. Inset left: enlargement of the boxed area. Inset right: we contrast one model (lower image) produced by evolution [see panel (a)] with a second model (upper image) that produces the same typical value of a but whose rates are uniformly scaled versions of the original model.

bound, the long-time limits of (1) and (3), respectively. Initially, the reference model is the original model, $\tilde{W}_{xy} = W_{xy}$, and so $a = a_0$ and $J_0 = 0$.

To perform an evolutionary step, we create a mutant model whose rates are

$$\hat{W}_{xy} = \mathrm{e}^{\epsilon(\eta_{xy}-1/2)}\,\tilde{W}_{xy}. \tag{5}$$

Here, ϵ is an evolutionary rate and η_{xy} is a uniformly distributed random number on (0, 1]. The parameter ϵ is a learning rate, and its effect is similar to other types of learning rates in machine learning, or basic step size in Monte Carlo simulation: if it is too small, then we do not explore parameter space rapidly enough; if it is too large, then the acceptance rate is too low; and somewhere in between these extremes its precise numerical value does not matter. The latter regime must be determined empirically, and we found values of ϵ of order 0.1 to be acceptable.

With this new set of rates, we run a new trajectory and compute the new values of a and J_0, called $\hat{a}$ and $\hat{J}_0$, respectively. If our selection criteria are fulfilled (see below), then we accept the mutation and set $\tilde{W}_{xy} = \hat{W}_{xy}$, $a = \hat{a}$, and $J_0 = \hat{J}_0$ (i.e., the mutant model becomes the new reference model); if not, we retain the current reference model.

We imposed two types of selection criteria. For the first, called a-evolution, we accepted the mutation if $\hat{a}$ is closer than a to a specified target value $a^\star$, i.e., if

$$|\hat{a} - a^\star| < |a - a^\star|. \tag{6}$$

For the second, called J-evolution, we accept the mutation if $\hat{J}_0$ is smaller than J_0 *and* if $\hat{a}$ lies within a tolerance δ of a specified pinning value $a^\dagger$, i.e., if

$$\hat{J}_0 < J_0 \quad \text{and} \quad |\hat{a} - a^\dagger| < \delta. \tag{7}$$

The process of a-evolution leads to reference models able to generate values of a far from a_0, while J-evolution leads to reference models that generate values of a in a manner as close as possible to the original model. The role of the parameter δ in (7) is to constrain the reference model to a particular window of a, and its value can be chosen for convenience (e.g., to ensure that we plot particular points along the rate function).

We alternated five steps of a-evolution, using an evolutionary rate of $\epsilon = 0.1$, with 50 steps of J-evolution, using an evolutionary rate of $\epsilon = 0.05$ and a tolerance of $\delta = 0.1$. During J-evolution, we chose the pinning value $a^\dagger$ to be the last value of a produced by the preceding phase of a-evolution. Upon reaching a specified value $a^\star$, we carried out an additional $N_{\mathrm{ev}} = 10^5$ steps of J evolution (with $a^\dagger = a^\star$), again using $\epsilon = 0.05$ and $\delta = 0.1$. We took N_{ev} large enough that the bound had stopped evolving under J-evolution. For the 4-state model, the chosen value 10^5 is much larger than necessary, because the bound stopped evolving after a few hundred steps. We carried out 100 independent simulations, each with a different target value $a^\star$.

Some of the models produced in this way are shown in Fig. 1(a), and the associated rate-function bounds are shown as green circles in panel (b). All points $(\tilde{a}_0, J_0)$ on the bound, derived from the typical behavior of the reference models, lie on the exact rate function of the original model, indicating that each reference model's typical dynamics is equivalent to the conditioned rare dynamics of the original model. Some of the reference models so obtained are shown

in panel (a) and in the inset of panel (b). The blue and gray dashed lines are, respectively, the Conway–Maxwell–Poisson bound[52] (see Appendix B) and the universal current bound.[50,51]

IV. A NEURAL-NETWORK ANSATZ FOR MODELS WITH LARGE STATE SPACES

A. A lattice model whose state space is small enough to diagonalize

In Sec. III, we saw that evolutionary reinforcement learning using 12 trainable parameters (the 12 rates of the reference model) permits accurate computation of the rate function, i.e., accurate computation of probabilities exponentially small in the trajectory length T. However, direct application of rate-based evolution is impractical for models with a large number of rates. To overcome this problem, we can encode the rates of the reference model as a neural network, and we illustrate this procedure in this section using the one-dimensional Fredrickson–Andersen (FA) model.[79]

The FA model is a lattice model with dynamical rules that give rise to slow relaxation and complex space–time behavior.[80] On each site i of a lattice of length L lives a spin S_i, which can be up (+1) or down (−1). Up-spins (respectively, down-spins) flip down (respectively, up) with rate $1-c$ (respectively, c) if at least one of their neighboring spins is up; if not, then they cannot flip. We take the dynamical observable a to be the number of configuration changes per unit time, $\alpha_{xy}=1$, often called activity.[45] To determine the large-deviation rate function $J(a)$ for activity, we chose a reference-model parameterization

$$\tilde{W}_{xy} = W_{xy} e^{w_0} e^{f_y - f_x}. \tag{8}$$

Here, w_0 is a parameter that effectively speeds up or slows down the clock,[54,81] and f_x is the value in state x of the neural network shown in Fig. 2. This network is inspired by the convolutional neural networks used to recognize images[82,83] and consists of a set of feature detectors or spin "filters" that scan the lattice for specified spin patterns. Here, we consider filters called k^α, each having L hidden nodes; the output of a hidden node is 1 if the k consecutive spins to which it is attached are all in state α, and is zero otherwise (i.e., the activation function is a step function). The network has one hidden layer. The weights connecting the input layer (the lattice) to the hidden layer are unity, and the weights connecting the hidden layer to the output node are denoted $w_k^\pm$; these are the trainable parameters of the network. All weights within a filter have the same value, a constraint suggested by the translational invariance of the model. The output of the network is

$$f_x = w_1 g_1(\boldsymbol{S}_x) + \sum_{k=2}^{K} \sum_{\alpha=\pm 1} w_k^\alpha g_k^\alpha(\boldsymbol{S}_x), \tag{9}$$

where $\boldsymbol{S}_x$ is the configuration of the lattice in state x, and $g_k^\alpha(\cdot)$ returns the number of active hidden nodes in the filter k^α [see Fig. 2(a)]. The reference model contains $2K$ trainable parameters: w_0, w_1 (only one type of 1-spin filter is necessary), and $w_2^\pm, \cdots, w_K^\pm$; note that $K=L$ when all filter types are used.

The form of (9) is similar to the multi-parameter auxiliary potential of Ref. 23, used to improve the convergence of the cloning method[48] in order to calculate the large-deviation function of the FA model. The present approach is different, however, in that the calculation is done using direct simulation of a reference model whose parameters are determined by an evolutionary process (rather than using rare-event algorithms such as cloning or transition-path sampling[84]), and results in the calculation of $J(a)$ directly, rather than its Legendre transform.[44]

To test the method, we considered the FA model with periodic boundary conditions and the parameter choices $c=0.3$ and $L=15$, the latter value being small enough that the exact $J(a)$ can be determined by diagonalization of the model's rate matrix; that function is shown as a black dashed line in Fig. 3(a). We next introduce the

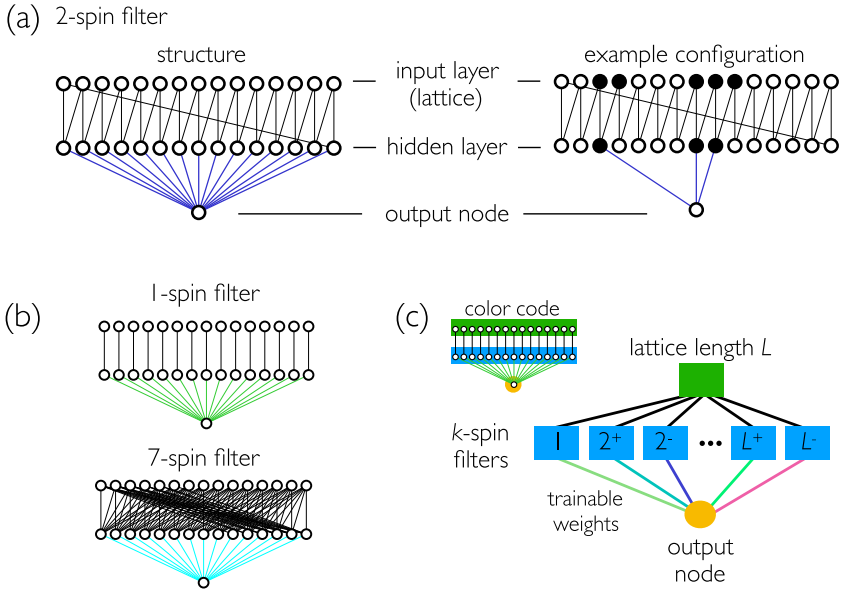

FIG. 2. Sketch of the neural-network reference-model ansatz used to compute the dynamical large deviations of the FA lattice model. (a) The building block of the network is a k^α-spin "filter," whose hidden nodes activate when the k consecutive spins to which they are attached are all of type $\alpha=\pm$ (periodic boundaries account for the diagonal line between input and hidden layers). Here, we show a 2^+-spin filter applied to a lattice of $L=15$ sites; shown right is an example configuration. The output of the filter is the number of hidden nodes that are on (here 3) multiplied by the weights denoted by the blue lines. (b) Structure of a 1-spin filter and a 7-spin filter (lattice size $L=15$). (c) The complete network contains a single hidden layer of $2K$ filters ($K \le L$), with $2K-1$ trainable parameters (the colored lines). The network output is the function f displayed in (9).

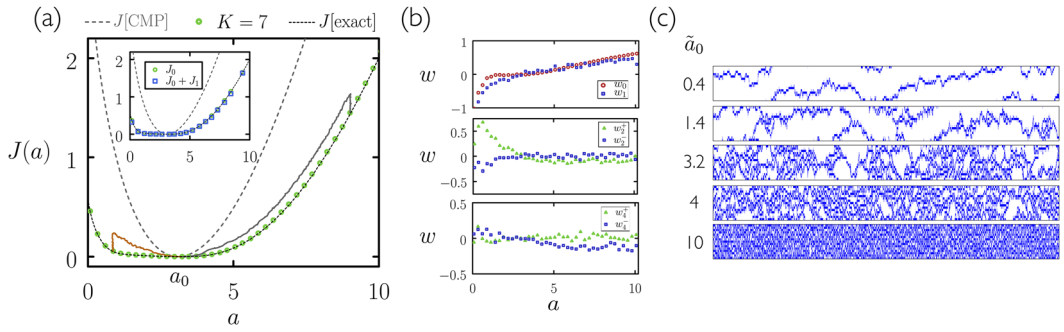

FIG. 3. (a) Evolutionary reinforcement learning using neural-network spin filters up to order $K = 7$ (green circles) closely approximates the large-deviation rate function $J(a)$ for activity a in the FA model of $L = 15$ sites (black). Also shown are the CMP universal activity bound[52] (gray dashed) and evolutionary trajectories of two neural-network reference models (gray and orange). The CMP bound is informative because it results from a set of reference models whose rates are uniform multiples of those of the original model and which therefore visit the same typical set of microstates as the original model, just faster or slower (see Appendix B); as a result, the discrepancy between the CMP bound and the true rate function indicates the importance of rare microstates to the rare behavior of the model. Inset: the numerically sampled rate function (blue squares) calculated from the bound plus correction term agrees with the exact answer. (b) Values of some of the weights of the neural network (9) for the reference models that produce the green circles in panel (a). (c) Space (vertical) vs time (horizontal) plots for trajectories of length $T = 2 \times 10^3$ for five different reference models. Blue pixels indicate up-spins. The typical values of the activity for each model are shown left of the plot; the center reference model is the original model.

reference model (8) and do evolutionary reinforcement learning on the weights of the network, as follows:

All neural-network weights $w \in \{w_0, w_1, \{w_K^\alpha\}\}$ of the reference model (8) were initially zero. Each proposed evolutionary move consisted of a shift of each weight by independent Gaussian-distributed random numbers of zero mean and variance $\sigma^2 = 10^{-4}$,

$$w \to w + \mathcal{N}(0, \sigma^2). \tag{10}$$

The parameter σ is a learning rate, and its effect is similar to other types of learning rates in machine learning, or basic step size in Monte Carlo simulation. We found values of σ of order 0.01 to be acceptable. We ran trajectories for $N = 10^5$ events and recorded the values of (1) and (3) after each proposed trajectory. We did a-evolution on the parameters w_0 and w_1 until a specified value $a^\star$ was reached. This procedure was as described for the 4-state model, with the additional restriction that the new bound must be not more than a value $\mu = 0.2$ larger than the current bound. That is, the proposed set of weights was accepted if

$$|\hat{a} - a^\star| < |a - a^\star| \quad \text{and} \quad \hat{J}_0 < J_0 + \mu. \tag{11}$$

We introduced the parameter μ in order to test the effect of replacing the alternating a- and J-evolution of Sec. III with a "regularized" form of a-evolution (one that does not allow the bound to grow beyond a particular size in any one step). If μ was chosen very small (e.g., $\lesssim 10^{-3}$), then a-evolution could not get going at all, because the bound must be allowed to increase in size at some point in the calculation. If μ was set very large (e.g., of order 10, so that the second requirement in (11) was effectively not present), then we observed the effect seen with the gray line in Fig. 5, whereby the bound obtained after a-evolution and prior to J-evolution was much larger than the exact value of J. For intermediate values of μ, such as the value 0.2 chosen here, the bound obtained prior to J-evolution was in general close to the exact answer (see the gray and orange lines in Fig. 3). However, the total computer time required in the cases of moderate and large μ was similar.

We then did J-evolution using a tolerance of $\delta = 0.02$ [see Eq. (7)], for $N_{\text{ev}} = 3 \times 10^4$ proposed trajectories, with higher-order spin filters applied. We ran 50 simulations, each with a different target value of a.

In Fig. 3(a), we show results of these calculations using spin filters up to order $K = 7$. Increasing K from 0 improves the quality of the bound until, for $K \gtrsim 4$, the bound becomes numerically close to the exact answer; see Fig. 4. This figure demonstrates that the quality of the bound exceeds that of the few-parameter, physically motivated ansatz used in Ref. 54. The neural network contains many fewer parameters than the model has rates (unlike in many deep-learning studies), and so we do not necessarily expect the bound to be exact. If the bound is good, the exact answer can be calculated

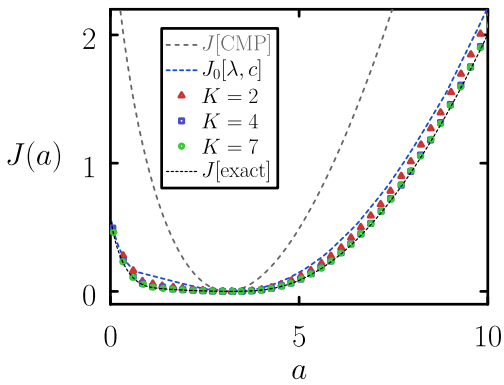

FIG. 4. As Fig. 3(a), showing results for neural-network spin filters up to order $K = 2, 4, 7$. For $K \gtrsim 4$, the bound is numerically close to the exact answer. Also shown is the CMP universal activity bound[52] (gray), which results from the typical dynamics of a reference model whose rates are uniform multiples of those of the original model, and the (c, λ) bound of Ref. 54 (blue). The latter is essentially equivalent to the case $K = 1$.

by computing a correction term.[54] We calculated the correction term using the neural-network reference model plus the procedure and *Python* script[85] described in Ref. 54 and show the bound plus converged correction in the inset to panel (a) of Fig. 3. Error bars are smaller than symbols. Here, the correction term is very small (which is apparent from a comparison of the bound, the green circles, and the exact answer, the black dashed line), indicating that the typical dynamics of this set of reference models is similar to the conditioned rare behavior of the original model. Comparison of these results with the exact result, and with the (c, λ)-bound from Ref. 54 (Fig. 4), indicates that rare trajectories of the FA model with parameter c resemble the typical trajectories of versions of the FA model with different values of the parameter c, but with slightly different tendencies to display spin domains of different lengths. These tendencies are quantified by the weights of the neural network, some of which are shown in Fig. 3(b). In panel (c), we show space–time plots of the trajectories of five reference models.

In Fig. 4, we reproduce some of the results shown in Fig. 3(a), together with results obtained for different values of K. In physical terms, the value of K determines the length scale over which the dynamical rules of the reference model act. The original model possesses only nearest-neighbor dynamical rules; its dynamics conditioned upon certain values of a involves potentially long-range correlations.[45,86] Figure 4 shows how closely (in terms of probabilities) the typical dynamics of reference models whose dynamical rules possess K-spin correlations approximate this conditioned dynamics: $K \approx 4$ is sufficient to closely approximate the rate function.

B. Lattice models whose state space is too large to diagonalize

With proof of principle demonstrated using models whose state space is small enough to solve by matrix diagonalization, we show in Fig. 5 that bounds produced by evolutionary learning can closely approximate rate functions whose calculation requires state-of-the-art numerical methods. For this purpose, we chose the 100-site FA model of Ref. 53 (for $c = 0.1$), whose state space is large enough that state-of-the-art methods are needed to compute its large-deviation rate function. This FA model has open boundary conditions. We also considered the 36-site FA model of Ref. 23, which has periodic boundary conditions.

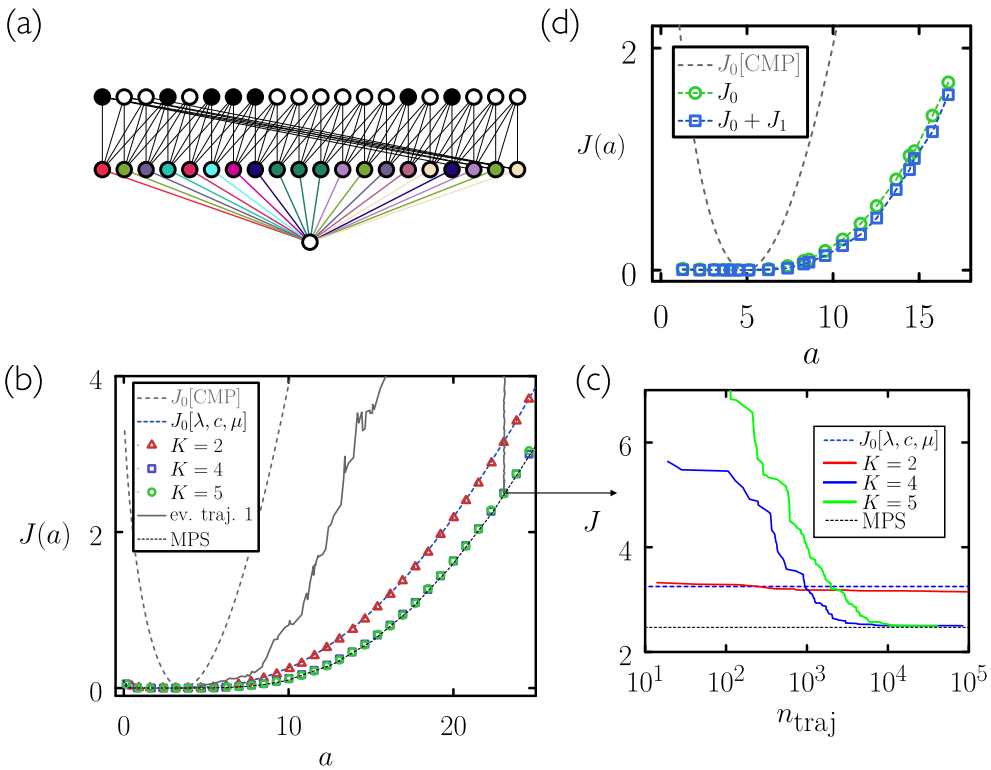

FIG. 5. (a) A spin filter (feature detector) related to those shown in Fig. 2, but generalized to recognize 2^K features in the vicinity of each lattice site (here, for the purpose of illustration, $L = 20$ and $K = 4$). Each hidden node possesses 2^K internal states (identified by colors) and the same number of parameters; the output of the network is (13). (b) Similar to Fig. 3(a), but using the FA model of Ref. 53 ($L = 100$, $c = 0.1$). We show the CMP bound[52] (gray), the three-parameter bound of Ref. 54 (blue), the results of evolutionary learning using the network shown in panel (a), for $K = 2$, 4, or 5, and the exact answer obtained using matrix product states[53] (black). Evolutionary learning produces a bound numerically close to the exact answer. We also show one evolutionary trajectory (gray). (c) Bound vs number of trajectories of J-evolution for one particular choice of a [labeled by the black arrow connecting panels (b) and (c)], for the cases $K = 2$, 4, and 5 [the latter corresponds to the gray line in panel (b)]. (d) Bound (green) and corrected bound (blue) for the FA model of Ref. 23 ($c = 0.2$, $L = 36$) using a neural-network reference model with $K = 4$.

Initial tests using the $L = 100$ FA model, done using evolutionary learning on the network shown in Fig. 2 (with $K = 5$), produced a bound that was visibly less close to the exact answer than in the case $L = 15$, suggesting the need for a neural-network ansatz able to detect more detailed features. We therefore replaced the network shown in Fig. 2 with the one shown in Fig. 5(a). This new network is capable of learning which features (spin patterns) are most significant; by contrast, the network shown in Fig. 2 searches only for homogeneous blocks of spins.

Each hidden node $i = 1, 2, \ldots L$ in the new network couples to K input nodes (lattice sites) and takes one of 2^K values. This value, called $h_x(i)$ in microstate x, is determined by the state of the K spins to which is it connected, via

$$h_x(i) = \sum_{m=0}^{K-1} 2^m \left(\frac{1 + S^x_{i+m}}{2} \right). \tag{12}$$

The output of the network in microstate x is then

$$f_x = \sum_{i=1}^{L} w_{h_x(i)}, \tag{13}$$

where the 2^K weights $w_{h_x(i)}$ are, along with w_0, the trainable parameters of the model. The reference-model ansatz is again (8), but now with (13) replacing (9).

We ran 40 evolutionary simulations, each with a different target value of a between 0.1 and 30 (the typical a of the original model is ~3.5). We turned on the neural network from the start and used trajectories of $N = 2 \times 10^5$ events. We did a-evolution [using Eq. (6)] to generate the desired values of a and then did J-evolution for $\approx 3 \times 10^4$ proposed trajectories. The results are shown in Fig. 5(b): for $K \gtrsim 4$, the bound produced is inexact, but numerically close to the exact answer. In panel (c), we show the evolution of the bound as a function of the number of evolutionary steps n_{traj}. As a guide to CPU consumption, 100 trajectories of $N = 2 \times 10^5$ events (each followed by a neural-network mutation step) take 5 s, 17 s, and 31 s for the cases $K = 2$, 4, and 5, respectively, on a 3.1 GHz Intel Core i7 processor (and so the total simulation time for the case $K = 5$ was of order 4 h on that processor).

In panel (d), we show the bound and converged corrected bound for the 36-site FA model of Ref. 23, providing further evidence that the strategy presented here can be applied to models whose treatment requires specialized methods.

We note that we used slightly different variants of the a- and J-evolution protocols for each of the 4-state model and the small and large FA models (each protocol is detailed above), in order to explore the effect of changing protocol. We did not find one protocol to be obviously better than the others, suggesting that a number of different evolutionary strategies can be used to tackle these problems.

V. CONCLUSIONS

In previous work, we showed how to calculate dynamical large-deviation rate functions using a variational ansatz for rare dynamics (VARD).[54] The first step of the VARD method is to calculate a rate-function bound, derived from the typical behavior of the ansatz, and we showed in that paper that ansätze containing a few parameters motivated by physical insight produced tight bounds for a set of models taken from the literature. In this paper, we have shown that multi-parameter ansätze, in the form of a relatively simple neural network ("VARDNet"), combined with evolutionary reinforcement learning, produce tighter bounds on dynamical large-deviation rate functions for a selection of models. In these cases, no physical insight into the model under study was required. Moving between ansätze containing 1 or 2 parameters, and ansätze containing 10 or 20 parameters, we move between bounds of similar quality to the universal bounds[50–52] and bounds that are numerically close to the exact rate function. Tight bounds provide a good starting point for the calculation of the exact rate function, or the study of the associated rare dynamics.

In treating the lattice models, we have introduced simple neural networks as reference-model ansätze for the rare behavior of each. The question of which network is best for a particular model and application is an open one. We used the single-layer architectures shown in Figs. 2 and 5, partly because the rates for reference-model spin flips then depend only on the states of the few feature detectors to which a spin is attached, and this allows relatively efficient and rapid updating of rate tables during the course of a continuous-time Monte Carlo simulation. A natural next step would be to apply a deeper network during later stages of evolution (see, e.g., Ref. 87, which appeared while this paper was in revision). Doing so would make for more costly simulation, but would allow each reference-model rate to be informed by the state of the entire lattice, thereby increasing the descriptive power of the ansatz.

The approach described here is a complement to the growing body of methods designed to compute large deviations.[21,23,24,44,45,48,49] It can also be straightforwardly adapted to treat other physical problems that involve time- or path-extensive quantities; one example is molecular self-assembly, whose outcome depends in a potentially complex way on the entire history of the interactions of a set of molecules.[88]

ACKNOWLEDGMENTS

We thank Hugo Touchette for the comments. This work was performed as part of a user project at the Molecular Foundry, Lawrence Berkeley National Laboratory, supported by the Office of Science, Office of Basic Energy Sciences, of the U.S. Department of Energy under Contract No. DE-AC02–05CH11231. D.J. acknowledges support from the Department of Energy Computational Science Graduate Fellowship. I.T. performed work at the National Research Council of Canada under the auspices of the AI4D Program. D.J. acknowledges support from the Department of Energy Computational Science Graduate Fellowship, under Contract No. DE-FG02–97ER25308.

APPENDIX A: LARGE DEVIATIONS BY CHANGE OF MODEL

For completeness, we present the derivation of Eq. (3) of the main text, which follows straightforwardly from the definition of the probability distribution. The derivation follows Ref. 54 with minor notational changes. For more on the ideas of dynamic importance sampling, see, e.g., Refs. 44, 61–67, and 21 (especially Sec. V).

Consider a continuous-time dynamics on a set of discrete states, defined by the master equation[89]

$$\partial_t P_x(t) = \sum_{y\neq x} W_{yx}P_y(t) - R_x P_x(t). \tag{A1}$$

Here, $P_x(t)$ is the probability that the system is in (micro)state x at time t, W_{xy} is the rate for passing from state x to state y, and $R_x = \sum_{y\neq x} W_{xy}$ is the escape rate from x. A standard way of simulating (A1) is as follows:[56] from state x, choose a new state y with probability

$$p_{xy} = \frac{W_{xy}}{R_x}, \tag{A2}$$

and a time increment Δt from the distribution

$$p_x(\Delta t) = R_x e^{-R_x \Delta t}. \tag{A3}$$

The dynamics defined by (A2) and (A3) generates a trajectory $\omega = x_0 \to x_1 \to \cdots \to x_{N(\omega)}$ consisting of $N(\omega)$ jumps $x_n \to x_{n+1}$ and associated jump times Δt_n. Associated with an ensemble of trajectories of length T is the probability distribution

$$\rho_T(A) = \sum_\omega p(\omega)\delta(T)\delta(A) \tag{A4}$$

of a time-extensive dynamical observable

$$A(\omega) = \sum_{n=0}^{N(\omega)-1} \alpha_{x_n x_{n+1}}. \tag{A5}$$

In these expressions, $\delta(X) \equiv \delta(X(\omega) - X)$ specifies a constraint on the trajectory, α_{xy} is the change of A upon moving from x to y, and $A(\omega)$ is the sum of these quantities over a single trajectory ω. We define $a(\omega) \equiv A(\omega)/T(\omega)$ as the time-intensive version of A. $T(\omega)$ is the elapsed time of trajectory ω, and $p(\omega)$ is the probability of a trajectory ω, proportional to a product of factors (A2) and (A3) for all jumps of the trajectory.

Fluctuations of a are quantified by $\rho_T(A)$, which for large T often adopts a large-deviation form[44,46]

$$\rho_T(A) \approx e^{-TJ(a)}. \tag{A6}$$

Direct evaluation of (A4) using the dynamics (A2) and (A3) leads to good sampling of $J(a)$ near the typical value a_0, where $J(a_0) = 0$, and poor sampling elsewhere. To overcome this problem, we can introduce a reference dynamics,

$$\tilde{p}_{xy} = \frac{\tilde{W}_{xy}}{\tilde{R}_x}, \tag{A7}$$

and

$$\tilde{p}_x(\Delta t) = \tilde{R}_x e^{-\tilde{R}_x \Delta t}, \tag{A8}$$

in which $\tilde{W}_{xy}$ is a modified version of the rate of the original model, and $\tilde{R}_x \equiv \sum_y \tilde{W}_{xy}$. Let $\tilde{p}(\omega)$ be the trajectory weight of the reference dynamics, proportional to a product of factors (A7) and (A8) for all jumps of the trajectory. We write

$$\langle\cdot\rangle^a \equiv \sum_\omega p(\omega)(\cdot)\delta(T)\delta(aT) \tag{A9}$$

and

$$\langle\cdot\rangle^a_{\rm ref} \equiv \sum_\omega \tilde{p}(\omega)(\cdot)\delta(T)\delta(aT) \tag{A10}$$

for the ensemble averages over trajectories (having length T and observable $A = aT$) of the original and reference models, respectively. We can then write (A4) as

$$\rho_T(A) = \langle 1\rangle^a \tag{A11}$$

$$= \langle e^{Tq(\omega)}\rangle^a_{\rm ref} \tag{A12}$$

$$= e^{T\langle q(\omega)\rangle^a_{\rm ref}} \langle e^{T\delta q(\omega)}\rangle^a_{\rm ref}. \tag{A12}$$

Here, $e^{Tq(\omega)} = p(\omega)/\tilde{p}(\omega)$ is the reweighting factor (also known as the likelihood ratio or Radon–Nikodym derivative[21,61]), and the quantity $\delta q(\omega) \equiv q(\omega) - \langle q(\omega)\rangle^a_{\rm ref}$; the angle brackets denote an average over many trajectories of a reference model with typical observable a.

We have

$$q(\omega) = T^{-1}\ln\frac{p(\omega)}{\tilde{p}(\omega)} = T^{-1}\sum_{n=0}^{N-1} q_{x_n x_{n+1}} + T^{-1}\Delta q_N, \tag{A13}$$

where

$$q_{x_n x_{n+1}} = \ln\frac{W_{x_n x_{n+1}}}{\tilde{W}_{x_n x_{n+1}}} - \Delta\tilde{t}_n(R_{x_n} - \tilde{R}_{x_n}), \tag{A14}$$

and $\Delta q_N \equiv -(T - \sum_{n=0}^{N-1}\Delta\tilde{t}_n)(R_{x_N} - \tilde{R}_{x_N})$. The latter term accounts for not leaving microstate x_N between the time of entry and the final time T. Here, $\Delta\tilde{t}_n = -\ln\eta/\tilde{R}_{x_n}$ is the jump time of the reference model (η is a random number uniformly distributed on (0, 1]).

Taking logarithms of (A12) and the large-T limit gives

$$J(\tilde{a}_0) = J_0(\tilde{a}_0) + J_1(\tilde{a}_0), \tag{A15}$$

where

$$J_0(\tilde{a}_0) = -\langle q(\omega)\rangle^{\tilde{a}_0}_{\rm ref} \tag{A16}$$

and

$$J_1(\tilde{a}_0) = -\frac{1}{T}\ln\langle e^{T\delta q(\omega)}\rangle^{\tilde{a}_0}_{\rm ref}. \tag{A17}$$

In these expressions, $\tilde{a}_0$ is the typical value of a for the reference model. The term (A16) is by Jensen's inequality an upper bound on the piece of the rate function $J(a)$ at the point $a = \tilde{a}_0$, i.e.,

$$J(\tilde{a}_0) \le J_0(\tilde{a}_0). \tag{A18}$$

The bound can be determined by computing the values of (A5) and (A13) for a suitably long reference-model trajectory. If the reference model's typical dynamics is similar to the conditioned rare dynamics of the original model (something we generally do not know in advance), then the bound $J_0(\tilde{a}_0)$ will be tight, and if it is tight enough, the exact value of $J(\tilde{a}_0)$ can be calculated from (A17) by sampling the (slightly) atypical behavior of the reference model.[54] In the main text, we show that evolutionary reinforcement learning can generate reference models for which the correction term is very small. The optimal reference model, called the driven or auxiliary process,[21,44,61] is one for which the bound is exact, meaning that its typical behavior is equivalent to the conditioned rare behavior of the original model.

APPENDIX B: THE CMP UNIVERSAL ACTIVITY BOUND

The Conway–Maxwell–Poisson (CMP) formula

$$J_{\mathrm{CMP}}(a) = \frac{k_0}{a_0}\left(a \ln \frac{a}{a_0} + a_0 - a\right) \tag{B1}$$

gives a bound on the large-deviation rate function $J(a)$ for any non-decreasing counting observable a; here, a_0 is the typical value of the observable, and k_0 is the typical dynamical activity k (the total number of configuration changes per unit time). Equation (B1) was derived in Ref. 52 from the result known as Level 2.5 of large deviations,[90,91] and we have used this form in Figs. 3–5 (for the case $a = k$).

We note here that the CMP formula can be straightforwardly derived from the generic bound (3), without using Level 2.5 of large deviations. Let a_0 and k_0 be the typical activities produced by an original model W_{xy}. Then, a reference model $\tilde{W}_{xy} = \gamma W_{xy}$, whose rates are uniformly rescaled versions of those of the original model, will produce typical activities γa_0 and γk_0 (a uniform rescaling of rates does not affect the choice of new state, i.e., $\tilde{W}_{xy}/\tilde{R}_x = W_{xy}/R_x$, and so the reference model will visit the same set of states as the original model, just faster or slower). The reference-model escape rate is then $\tilde{R}_x = \gamma R_x$. In (3), we assume the long-time, steady-state limit and so replace the fluctuating jump time $\Delta\tilde{t}_n$ with its mean $1/\tilde{R}_{x_n}$, giving

$$\begin{aligned} J_0(\tilde{a}_0) &= T^{-1}\sum_{n=0}^{N-1}\left(\ln\gamma + \frac{1-\gamma}{\gamma}\right) \\ &= \tilde{k}_0\left(\ln\frac{\tilde{a}_0}{a_0} + \frac{a_0 - \tilde{a}_0}{\tilde{a}_0}\right) \\ &= \frac{\tilde{k}_0}{\tilde{a}_0}\left(\tilde{a}_0 \ln\frac{\tilde{a}_0}{a_0} + a_0 - \tilde{a}_0\right) \\ &= \frac{k_0}{a_0}\left(\tilde{a}_0 \ln\frac{\tilde{a}_0}{a_0} + a_0 - \tilde{a}_0\right), \end{aligned} \tag{B2}$$

where $\tilde{a}_0 = \gamma a_0$ and $\tilde{k}_0 = \gamma k_0$ are, respectively, the typical values of a and k for the reference model (a_0 and k_0 are the analogous quantities for the original model). Equation (B2) is the bound associated with the single reference model whose rates are $\tilde{W}_{xy} = \gamma W_{xy}$. By choosing different values of γ, we create a family of reference models, each with a distinct typical behavior, and so we can replace $\tilde{a}_0$ in Eq. (B2) with the general a; doing so, we recover Eq. (B1), the CMP bound.

The derivation leading to (B2) specifies only that $a = A/T$ be derived from a time-extensive quantity A, so that rescaling all rates by a factor $\gamma > 0$ changes the typical value of the observable, a_0, to γa_0. Thus, the CMP bound applies to *any* time-extensive quantity, including currents, not just non-decreasing counting variables. This fact justifies its inclusion in Fig. 1, where we consider entropy production (a current). However, the CMP bound does not address the $a < 0$ sector, which cannot be accessed if the typical value of the observable of the original model is $a_0 > 0$ (because any reference model obtained under a rescaling of rates has $\tilde{a}_0 = \gamma a_0 > 0$).

A simple way to produce a bound pertaining to the $a < 0$ sector is to use the γ-rescaling on the *time-reversed* version of the original model. To see this, we proceed as follows: Consider the reference model obtained by rescaling the rates of the original model, W_{xy}, by the exponential of (minus) the entropy production,

$$\tilde{W}_{xy} = \mathrm{e}^{-\sigma_{xy}} W_{xy} \tag{B3}$$

$$= \frac{\pi_y}{\pi_x}\frac{p_{yx}}{p_{xy}} W_{xy} \tag{B4}$$

$$= \frac{R_x}{\pi_x}\pi_y p_{yx}. \tag{B5}$$

Here, we are using standard notation for Markov chains: $p_{xy} = W_{xy}/R_x$ is the probability of moving to (micro)state y, given that we are in state x; $R_x = \sum_y W_{xy}$ is the escape rate from x; and π_x is the invariant measure, which satisfies

$$\pi_x = \sum_y \pi_y p_{yx}. \tag{B6}$$

Summing (B5) over y and using (B6) shows that the escape rate of the reference model is equal to that of the original,

$$\tilde{R}_x = \sum_y W_{xy} = \sum_y (\mathrm{B5}) = \frac{R_x}{\pi_x}\pi_x = R_x. \tag{B7}$$

Then upon dividing (B4) by $\tilde{R}_x = R_x$, we have

$$\tilde{p}_{xy} = \frac{\pi_y}{\pi_x} p_{yx}, \tag{B8}$$

and so this reference model generates the time-reversed Markov chain.[92] If the observable a is a current, odd under time reversal, then the typical value of the observable in the reference model is $\tilde{a}_0 = -a_0$.

To determine the value of the bound associated with the time-reversed model, we insert (B3) into the long-time version of (3), giving

$$J_0 = -T^{-1}\sum_{n=0}^{N-1}\sigma_{x_n x_{n+1}} = -\tilde{\sigma}_0 = \sigma_0. \tag{B9}$$

Thus, choosing the time-reversed model to be the reference model gives as a bound a single point $(-a_0, \sigma_0)$ on the rate function of any current a; here, a_0 and σ_0 are the typical values of the current and the entropy production rate in the original model.

If we now apply a γ-rescaling to the time-reversed model, we create a family of reference models with rates

$$\tilde{W}_{xy} = \gamma \mathrm{e}^{-\sigma_{xy}} W_{xy}. \tag{B10}$$

Using (3) and the results (B2) and (B9), it is straightforward to show that the bound associated with this family of reference models is

$$J_0(a) = \frac{\sigma_0}{a_0}|a| + \frac{k_0}{a_0}\left(|a| \ln\frac{|a|}{a_0} + a_0 - |a|\right). \tag{B11}$$

Hence, one bound on any current a is provided by the combination of (B2) (with $\tilde{a}_0 \to a$) for $a \geq 0$ and (B11) for $a < 0$. We show this bound (for the choice $a = \sigma$ for the 4-state model) as a blue dotted line in Fig. 6. The double-well form results from the fact that the associated family of reference models is a glued-together combination of forward and time-reversed "original" models with uniformly rescaled rates. A comparison with the universal current bound[50,51] (gray dotted line) shows that the latter was derived from a different family of models (see Figs. 2 and 3 of Ref. 54 for a comparison between the universal current bound and the bounds produced by other families of reference models).

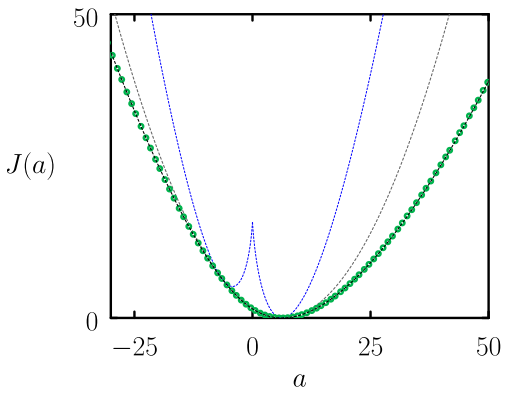

FIG. 6. Supplement to Fig. 1 of the main text, the large-deviation rate function $J(a)$ for entropy production a in a 4-state model (black and green). We show as a blue dashed line the CMP universal activity bound[52] and its time reverse, Eq. (B11), which together provide a rudimentary bound on any current. Shown in gray is the universal current bound.[50,51]

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

[1] J. Behler and M. Parrinello, Phys. Rev. Lett. **98**, 146401 (2007).
[2] K. Mills, M. Spanner, and I. Tamblyn, Phys. Rev. A **96**, 042113 (2017).
[3] A. L. Ferguson and J. Hachmann, Mol. Syst. Des. Eng. **3**, 429 (2018).
[4] N. Artrith, A. Urban, and G. Ceder, J. Chem. Phys. **148**, 241711 (2018).
[5] A. Singraber, T. Morawietz, J. Behler, and C. Dellago, J. Phys.: Condens. Matter **30**, 254005 (2018).
[6] C. Desgranges and J. Delhommelle, J. Chem. Phys. **149**, 044118 (2018).
[7] B. Thurston and A. Ferguson, Mol. Simul. **44**, 930 (2018).
[8] A. Singraber, J. Behler, and C. Dellago, J. Chem. Theory Comput. **15**, 1827 (2019).
[9] J. Han *et al.*, arXiv:1611.07422 (2016).
[10] K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, and A. Tkatchenko, Nat. Commun. **8**, 13890 (2017).
[11] K. Yao, J. E. Herr, D. W. Toth, R. Mckintyre, and J. Parkhill, Chem. Sci. **9**, 2261 (2018).
[12] K. T. Schütt, H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko, and K.-R. Müller, J. Chem. Phys. **148**, 241722 (2018).
[13] J. Carrasquilla and R. G. Melko, Nat. Phys. **13**, 431 (2017); arXiv:1605.01735.
[14] N. Portman and I. Tamblyn, J. Comput. Phys. **350**, 871 (2017).
[15] B. S. Rem, N. Käming, M. Tarnowski, L. Asteria, N. Fläschner, C. Becker, K. Sengstock, and C. Weitenberg, Nat. Phys. **15**, 917 (2019).
[16] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction* (MIT Press, 2018).
[17] T. P. I. Ahamed, V. S. Borkar, and S. Juneja, Oper. Res. **54**, 489 (2006).
[18] A. Basu, T. Bhattacharyya, and V. S. Borkar, Math. Oper. Res. **33**, 880 (2008).
[19] V. S. Borkar, in Proceedings of the 19th International Symposium on Mathematical Theory of Networks and Systems–MTNS, 2010, Vol. 5.
[20] V. S. Borkar, S. Juneja, A. A. Kherani *et al.*, Commun. Inf. Syst. **3**, 259 (2003).
[21] R. Chetrite and H. Touchette, J. Stat. Mech.: Theory Exp. **2015**, P12001.
[22] H. J. Kappen and H. C. Ruiz, J. Stat. Phys. **162**, 1244 (2016).
[23] T. Nemoto, R. L. Jack, and V. Lecomte, Phys. Rev. Lett. **118**, 115702 (2017).
[24] G. Ferré and H. Touchette, J. Stat Phys. **172**, 1525 (2018).
[25] T. A. Bojesen, Phys. Rev. E **98**, 063303 (2018).
[26] C. J. C. H. Watkins and P. Dayan, Mach. Learn. **8**, 279 (1992).
[27] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, arXiv:1312.5602 (2013).
[28] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, Nature **518**, 529 (2015).
[29] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, J. Artif. Intell. Res. **47**, 253 (2013).
[30] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, in *ICML'16: Proceedings of the 33rd International Conference on International Conference on Machine Learning* (Association for Computing Machinery, 2016), Vol. 48, pp. 1928–1937.
[31] Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. d. L. Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq *et al.*, arXiv:1801.00690 (2018).
[32] E. Todorov, T. Erez, and Y. Tassa, in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2012), pp. 5026–5033.
[33] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (John Wiley & Sons, 2014).
[34] A. Asperti, D. Cortesi, and F. Sovrano, "Crawling in rogue's dungeons with (partitioned) A3C," in *Machine Learning, Optimization, and Data Science*, Lecture Notes in Computer Science, edited by G. Nicosia, P. Pardalos, G. Giuffrida, (Springer, Cham, 2018), Vol. 11331.
[35] M. Riedmiller, *European Conference on Machine Learning* (Springer, 2005), pp. 317–328.
[36] M. Riedmiller, T. Gabel, R. Hafner, and S. Lange, Auton. Robots **27**, 55 (2009).
[37] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, arXiv:1707.06347 (2017).
[38] F. P. Such, V. Madhavan, E. Conti, J. Lehman, K. O. Stanley, and J. Clune, arXiv:1712.06567 (2017).
[39] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, arXiv:1606.01540 (2016).
[40] M. Kempka, M. Wydmuch, G. Runc, J. Toczek, and W. Jaśkowski, in *2016 IEEE Conference on Computational Intelligence and Games (CIG)* (IEEE, 2016), pp. 1–8.
[41] M. Wydmuch, M. Kempka, and W. Jaśkowski, arXiv:1809.03470 (2018).
[42] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, Nature **529**, 484 (2016).
[43] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, Nature **550**, 354 (2017).
[44] H. Touchette, Phys. Rep. **478**, 1 (2009).
[45] J. P. Garrahan, R. L. Jack, V. Lecomte, E. Pitard, K. van Duijvendijk, and F. van Wijland, J. Phys. A: Math. Theor. **42**, 075007 (2009).
[46] F. Den Hollander, *Large Deviations* (American Mathematical Soceity, 2008), Vol. 14.
[47] R. S. Ellis, *Entropy, Large Deviations, and Statistical Mechanics* (Springer, 2007).
[48] C. Giardina, J. Kurchan, and L. Peliti, Phys. Rev. Lett. **96**, 120603 (2006).
[49] U. Ray, G. K.-L. Chan, and D. T. Limmer, Phys. Rev. Lett. **120**, 210602 (2018).
[50] P. Pietzonka, A. C. Barato, and U. Seifert, Phys. Rev. E **93**, 052145 (2016).
[51] T. R. Gingrich, J. M. Horowitz, N. Perunov, and J. L. England, Phys. Rev. Lett. **116**, 120601 (2016).
[52] J. P. Garrahan, Phys. Rev. E **95**, 032134 (2017).
[53] M. C. Bañuls and J. P. Garrahan, Phys. Rev. Lett. **123**, 200601 (2019).
[54] D. Jacobson and S. Whitelam, Phys. Rev. E **100**, 052139 (2019).
[55] In Appendix B we show that the CMP bound of Ref. 52 is equivalent to a 1-parameter VARD bound.
[56] D. T. Gillespie, J. Phys. Chem. **81**, 2340 (1977).
[57] U. Seifert, Phys. Rev. Lett. **95**, 040602 (2005).
[58] T. Speck, A. Engel, and U. Seifert, J. Stat. Mech.: Theory Exp. **2012**, P12001.
[59] V. Lecomte, A. Imparato, and F. van Wijland, Prog. Theor. Phys. Suppl. **184**, 276 (2010).
[60] É. Fodor, M. Guo, N. S. Gov, P. Visco, D. A. Weitz, and F. van Wijland, EuroPhys. Lett. **110**, 48005 (2015).
[61] J. Bucklew, *Introduction to Rare Event Simulation* (Springer Science & Business Media, 2013).

[62]P. W. Glynn and D. L. Iglehart, Manage. Sci. **35**, 1367 (1989).
[63]J. S. Sadowsky and J. A. Bucklew, IEEE Trans. Inf. Theory **36**, 579 (1990).
[64]J. A. Bucklew, P. Ney, and J. S. Sadowsky, J. Appl. Probab. **27**, 44 (1990).
[65]J. A. Bucklew, *Large Deviation Techniques in Decision, Simulation, and Estimation* (Wiley, New York, 1990).
[66]S. Asmussen and P. W. Glynn, *Stochastic Simulation: Algorithms and Analysis* (Springer Science & Business Media, 2007), Vol. 57.
[67]S. Juneja and P. Shahabuddin, *Handbooks in Operations Research and Management Science* (Elsevier, 2006), Vol. 13, p. 291.
[68]A. Guyader and H. Touchette, arXiv:2003.05274 (2020).
[69]Y. Li and F. Cao, Eur. J. Oper. Res. **224**, 333 (2013).
[70]S. S. Singh, V. B. Tadić, and A. Doucet, Eur. J. Oper. Res. **178**, 808 (2007).
[71]J. H. Holland, Sci. Am. **267**, 66 (1992).
[72]D. B. Fogel and L. C. Stayton, BioSystems **32**, 171 (1994).
[73]J. Lehman, J. Chen, J. Clune, and K. O. Stanley, in *Proceedings of the Genetic and Evolutionary Computation Conference* (Association for Computing Machinery, 2018), pp. 450–457.
[74]T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever, arXiv:1703.03864 (2017).
[75]X. Zhang, J. Clune, and K. O. Stanley, arXiv:1712.06564 (2017).
[76]J. Lehman, J. Chen, J. Clune, and K. O. Stanley, in *Proceedings of the Genetic and Evolutionary Computation Conference* (Association for Computing Machinery, 2018), pp. 117–124.
[77]E. Conti, V. Madhavan, F. P. Such, J. Lehman, K. Stanley, and J. Clune, *Advances in Neural Information Processing Systems* (Curran Associates, 2018), pp. 5027–5038.
[78]The model's rates are $W_{12} = 3$, $W_{13} = 10$, $W_{14} = 9$, $W_{21} = 10$, $W_{23} = 1$, $W_{24} = 2$, $W_{31} = 6$, $W_{32} = 4$, $W_{34} = 1$, $W_{41} = 7$, $W_{42} = 9$, and $W_{43} = 5$.
[79]G. H. Fredrickson and H. C. Andersen, Phys. Rev. Lett. **53**, 1244 (1984).
[80]J. P. Garrahan and D. Chandler, Phys. Rev. Lett. **89**, 035704 (2002).
[81]In order to calculate the correction to the bound (3), the parameterization using the variable called λ in Ref. 54 is more efficient; to calculate the bound itself the choice (w_0 or λ) makes little difference.
[82]A. Krizhevsky, I. Sutskever, and G. E. Hinton, *Advances in Neural Information Processing Systems* (Curran Associates, Inc., 2012), pp. 1097–1105.
[83]Y. LeCun, L. Bottou, Y. Bengio, P. Haffner *et al.*, Proc. IEEE **86**, 2278 (1998).
[84]P. G. Bolhuis, D. Chandler, C. Dellago, and P. L. Geissler, Annu. Rev. Phys. Chem. **53**, 291 (2002).
[85]D. Jacobson and S. Whitelam, vard_correction, https://github.com/drdrjacobs/vard_correction.git, 2019.
[86]T. Bodineau, V. Lecomte, and C. Toninelli, J. Stat. Phys. **147**, 1 (2012).
[87]T. H. E. Oakes, A. Moss, and J. P. Garrahan "A deep learning functional estimator of optimal dynamics for sampling large deviations," Mach. Learn.: Sci. Technol. (to be published).
[88]S. Whitelam and I. Tamblyn, Phys. Rev. E **101**, 052604 (2020).
[89]K. Binder, *Monte Carlo Methods in Statistical Physics* (Springer, 1986), pp. 1–45.
[90]C. Maes and K. Netočný, Europhys. Lett. **82**, 30003 (2008).
[91]L. Bertini, A. Faggionato, D. Gabrielli *et al.*, *Annales de l'Institut Henri Poincaré, Probabilités et Statistiques* (Institut Henri Poincaré, 2015), Vol. 51, pp. 867–900.
[92]W. K. Hastings, *Monte Carlo Sampling Methods Using Markov Chains and Their Applications* (Oxford University Press, 1970).