


Learning to grow: Control of material self-assembly using evolutionary reinforcement learning

Stephen Whitelam*

*Molecular Foundry, Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley, California 94720, USA*Isaac Tamblyn[†]*National Research Council of Canada, Ottawa, Ontario, Canada
and Vector Institute for Artificial Intelligence, Toronto, Ontario, Canada* (Received 29 December 2019; revised manuscript received 7 February 2020; accepted 29 March 2020; published 11 May 2020)

We show that neural networks trained by evolutionary reinforcement learning can enact efficient molecular self-assembly protocols. Presented with molecular simulation trajectories, networks learn to change temperature and chemical potential in order to promote the assembly of desired structures or choose between competing polymorphs. In the first case, networks reproduce in a qualitative sense the results of previously known protocols, but faster and with higher fidelity; in the second case they identify strategies previously unknown, from which we can extract physical insight. Networks that take as input the elapsed time of the simulation or microscopic information from the system are both effective, the latter more so. The evolutionary scheme we have used is simple to implement and can be applied to a broad range of examples of experimental self-assembly, whether or not one can monitor the experiment as it proceeds. Our results have been achieved with no human input beyond the specification of which order parameter to promote, pointing the way to the design of synthesis protocols by artificial intelligence.

DOI: [10.1103/PhysRevE.101.052604](https://doi.org/10.1103/PhysRevE.101.052604)**I. INTRODUCTION**

Molecular self-assembly is the spontaneous organization of molecules or nanoparticles into ordered structures [1–6]. It is a phenomenon that happens out of equilibrium, and so while we have empirical and theoretical understanding of certain self-assembling systems and certain processes that occur during assembly [7–20], we lack a predictive theoretical framework for self-assembly. That is, given a set of molecules and ambient conditions, and an observation time, we cannot in general predict which structures and phases the molecules will form and what will be the yield of the desired structure when (and if) it forms. As a result, industrial processes that use self-assembly, such as the crystallization of pharmaceuticals, require an empirical search of materials and protocols, often at considerable time and cost [21–24].

Absent a theoretical framework for self-assembly, an alternative is to seek assistance from machine learning in order to attempt to control self-assembly without human intervention. In this paper we show that neural-network-based evolutionary reinforcement learning can be used to develop protocols for the control of self-assembly, without prior understanding of what constitutes a good assembly protocol. Reinforcement learning is a branch of machine learning concerned with learning to perform actions so as to achieve an objective [25], and has been used recently to play computer games better than humans can [26–43]. Neuroevolution [43–50] is an approach

to reinforcement learning that is much less widely applied than value-based methods [25], but is a simple and powerful method that is naturally suited to “sparse-reward” problems such as self-assembly, where the outcome of assembly (good or bad) is not always apparent until its latter stages. Here we apply neuroevolutionary learning to stochastic molecular simulations of patchy particles, a standard choice for representing anisotropically interacting molecules, nanoparticles, or colloids [10,51–57]. While a neural network cannot influence the fundamental dynamical laws by which such systems evolve [58], it can control the parameters that appear in the dynamical algorithm, such as temperature, chemical potential, and other environmental conditions. In this way the network can influence the sequence of microstates visited by the system. We show that a neural network can learn to enact a time-dependent protocol of temperature and chemical potential (called a *policy* in reinforcement learning) in order to promote the self-assembly of a desired structure, or choose between two competing polymorphs. In both cases the network identifies strategies different to those informed by human intuition, but which can be analyzed and used to provide new insight. We use networks that take only elapsed time as their input, and networks that take microscopic information from the system. Both learn under evolution, and networks permitted microscopic information learn better than those that are not.

Networks enact protocols that are out of equilibrium, in some cases far from equilibrium, and so are not well described by existing theories. These “self-assembly kinetic yield” networks act to promote a particular order parameter for self-assembly at the end of a given time interval, with no

*swhitelam@lbl.gov

†isaac.tamblyn@nrc.ca

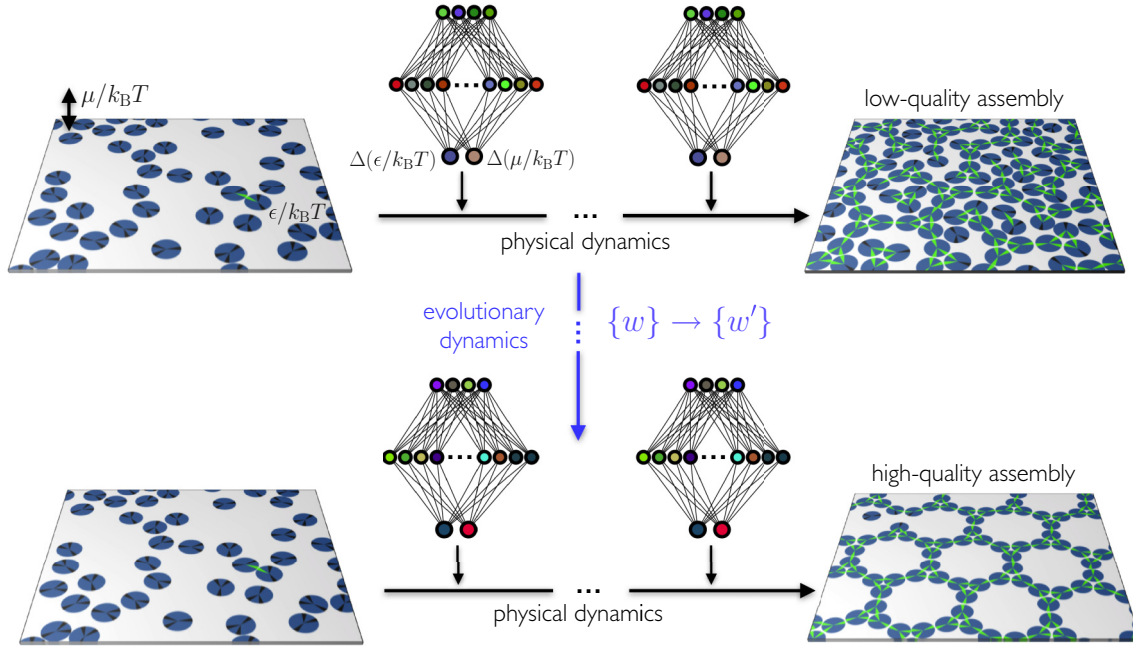


FIG. 1. In this paper we show that neural-network policies trained by evolutionary reinforcement learning can enact efficient time- and configuration-dependent protocols for molecular self-assembly. A neural network periodically controls certain parameters of a system, and evolutionary learning applied to the weights of a neural network (indicated as colored nodes) results in networks able to promote the self-assembly of desired structures. The protocols that give rise to these structures are then encoded in the weights of a self-assembly kinetic yield net.

consideration for whether a process results in an equilibrium outcome or not. It is therefore distinct from feedback approaches designed to promote near-equilibrium behavior [59]. Our approach is similar in intent to Ref. [60], in which dynamic programming is used to find protocols able to promote colloidal crystallization using an external field. One important difference between that work and ours is that our scheme does not require measurement of the order parameter we wish to promote (except at the end of the experiment), making it applicable to molecular and nanoscale systems whose microscopic states cannot be observed as they evolve. We also use a neural network to encode the assembly protocol rather than a model of discretized states. Our approach is also similar to that of Ref. [61] in the sense that we use evolutionary search to optimize assembly, but we allow the learning procedure to respond to both temporal and microscopic information via the use of a neural network. Our approach is complementary to efforts that use machine learning to analyze existing self-assembly pathways [62,63] or to infer or design structure-property relationships for self-assembling molecules [64–66]. The present scheme is simple and can be straightforwardly altered to observe an arbitrary number of system features, and to control an arbitrary number of system parameters, and so can be applied to a wide range of experimental systems.

In Sec. II we describe the evolutionary scheme, which involves alternating physical and evolutionary dynamics. In Sec. III we show that it leads to networks able to promote the self-assembly of a certain structure faster and better than intuitive cooling protocols can. In Sec. IV we show that networks can learn to select between two polymorphs that are equal in energy and that form in unpredictable quanti-

ties under slow cooling protocols. The strategy used by the networks to achieve this selection provides new insight into the self-assembly of the system under study. We conclude in Sec. V. Networks learn these efficient and new self-assembly protocols with no human input beyond the specification of which target parameter to promote, pointing the way to the design of synthesis protocols by artificial intelligence.

II. EVOLUTIONARY REINFORCEMENT LEARNING OF SELF-ASSEMBLY PROTOCOLS

We sketch in Fig. 1 an evolutionary scheme by which a self-assembly kinetic yield net can learn to control self-assembly. We consider a computational model of molecular self-assembly, patchy disks of diameter a on a two-dimensional square substrate of edge length $50a$. The substrate (simulation box) possesses periodic boundary conditions in both directions. Disks, which cannot overlap, are minimal representations of molecules, and the patches denote their ability to make mutual bonds at certain angles. By choosing certain patch angles, widths, and binding-energy scales it is possible to reproduce the dynamic and thermodynamic behavior of real molecular systems of a broad range of length scales and material types [56]. The disk model is a good system on which to test the application of evolutionary learning to self-assembly, because it is simple enough to simulate for long times, and its behavior is complex enough to capture several aspects of real self-assembly, including the formation of competing polymorphs and structures that are not the thermodynamically stable one. Choosing protocols to promote the formation of particular structures within the disk

model is therefore nontrivial and serves as a meaningful test of the learning procedure.

Two disks receive an energetic reward of $-\epsilon/k_B T$ if their center-to-center distance r is between a and $a + a/10$ and if the line joining those disks cuts through one patch on each disk [67]. In addition, we sometimes require patches to possess certain identities in order to bind, mimicking the ability of, e.g., DNA to be chemically specific [68]. In this paper we consider disk types with and without DNA-type specificity. Bound patches are shown green in figures, and unbound patches are shown black. In figures we often draw the convex polygons formed by joining the centers of bound particles [56]. Doing so makes it easier to spot regions of order by eye. Polygon counts serve as a useful order parameter for self-assembly, because they are related (in some cases proportional) to the number of unit cells of the desired material. We denote by N_α the number of convex α -gons within a simulation box.

We simulated this system in order to mimic an experiment in which molecules are deposited on a surface and allowed to evolve. We use two stochastic Monte Carlo algorithms to do so. One is a grand-canonical algorithm that allows disks to appear on the substrate or disappear into a notional solution [58]; the other is the virtual-move Monte Carlo algorithm [69,70] that allows disks to move collectively on the surface in an approximation of Brownian motion [71]. If M is the instantaneous number of disks on the surface, then we attempt virtual moves with probability $M/(1 + M)$ and attempt grand-canonical moves otherwise. Doing so ensures that particle deposition occurs at a rate (for fixed control parameters) that is roughly insensitive of substrate density. The acceptance rates for grand-canonical moves are given in Ref. [56] (essentially the textbook rates [58] with the replacement $M \rightarrow M + 1$ to preserve detailed balance in the face of a fluctuating proposal rate). One such decision constitutes one Monte Carlo step [72].

The grand-canonical algorithm is characterized by a chemical potential $\mu/k_B T$, where $k_B T$ is the energy scale of thermal fluctuations. Positive values of this parameter favor a crowded substrate, while negative values favor a sparsely occupied substrate. If the interparticle bond strength $\epsilon/k_B T$ is large, then there is, in addition, a thermodynamic driving force for particles to assemble into structures. (In experiment, bond strength can be controlled by different mechanisms, depending on the physical system, including temperature or salt concentration; here, for convenience, we sometimes describe increasing $\epsilon/k_B T$ as “cooling,” and decreasing $\epsilon/k_B T$ as “heating.”) For fixed values of these parameters the simulation algorithm obeys detailed balance, and so the system will evolve toward its thermodynamic equilibrium. Depending on the parameter choices, this equilibrium may correspond to an assembled structure or to a gas or liquid of loosely associated disks. For finite simulation time there is no guarantee that we will reach this equilibrium. Here we consider evolutionary simulations or trajectories of $t_0 = 10^9$ individual Monte Carlo steps (not sweeps, or steps per particle), starting from substrates containing 500 randomly placed nonoverlapping disks. These are relatively short trajectories in self-assembly terms: The slow cooling protocols of Ref. [68] used trajectories about 100 times longer.

Each trajectory starts with control-parameter values $\epsilon/k_B T = 3$ and $\mu/k_B T = 2$, which does not give rise to self-assembly. As a trajectory progresses, a neural network chooses, every $10^{-3}t_0$ Monte Carlo steps, a change $\Delta(\mu/k_B T)$ and $\Delta(\epsilon/k_B T)$ of the two control parameters of the system (and so the same network acts 1000 times within each trajectory). These changes are added to the current values of the relevant control parameter, as long as they remain within the intervals $\epsilon/k_B T \in [0, 20]$ and $\mu/k_B T \in [-20, 20]$ (if a control parameter moves outside of its specified interval then it is returned to the edge of the interval). Between neural-network actions, the values of the control parameters are held fixed. Networks are fully connected architectures with 1000 hidden nodes and two output nodes, and a number of input nodes appropriate for the information they are fed. We used tanh activations on the hidden nodes; the full network function is given in the Appendix.

Training of the network is done by evolution [43]. We run 50 initial trajectories, each with a different, randomly initialized neural network. Each network’s weights and biases $\{w\}$ are independent Gaussian random numbers of zero mean and unit variance. The collection of 50 trajectories produced by this set of 50 networks is called generation 0. After these trajectories run we assess each according to the number N_α of convex α -gons present in the simulation box; the value of α depends on the disk type under study and the structure whose assembly we wish to promote. The five networks whose trajectories have the largest values of N_α are chosen to be the “parents” of generation 1. Generation 1 consists of these five networks, plus 45 mutants. Mutants are made by choosing at random one of the parent networks and adding to each weight and bias a Gaussian random number of zero mean and variance 0.01. After simulation of generation 1 we repeat the evolutionary procedure in order to create generation 2. Alternating the physical dynamics (the self-assembly trajectories) and the evolutionary dynamics (the neural-network weight mutation procedure) results in populations of networks designed to control self-assembly conditions so as to promote certain order parameters.

Each evolutionary scheme used one of three types of network. The first, called the *time network* for convenience, has a single input node that takes the value of the scaled elapsed time of the trajectory, $t/t_0 \in [0, 1]$. The second, called the *microscopic network* for convenience, has $P + 1$ input nodes, where P is the number of patches on the disk. Input node $i \in \{0, 1, \dots, P\}$ takes the value S_i , the number of particles in the simulation box that possess i engaged patches (divided by 1000). The third neural network type has $P + 2$ input nodes and takes both t/t_0 and the S_i as inputs. We chose the time network so as to explore the ability of a network to influence the self-assembly protocol if it cannot observe the system at all. We chose the microscopic network to see if a network able to observe the system can do better than one that cannot. We do not intend for its input to be a precise analog of an experimental measurement, but there are several experimental techniques able to access similar information, such as the averaged number of particles in certain types of environment, or the approximate degree of aggregation present in a system [13].

It is important to note that these microscopic inputs are not related in a simple way to the evolutionary parameters N_α , the

number of convex α -gons in the box, that we wish to optimize. For instance, in Sec. III, both dense disordered networks (with small values of N_{12}) and well-assembled structures (with large values of N_{12}) can contain similar numbers of maximally coordinated particles. In Sec. IV, the two polymorphs we ask a network to choose between, one described by N_6 and the other by N_4 , have identical coordination numbers. Thus a network must learn the connection between the data it is fed and the evolutionary order parameters we aim to maximize. Our intent was to mimic an experiment in which which some microscopic information about a system is available, but the quality of assembly can only be assessed after the experiment has run to completion. The success of the learning scheme in the absence of any system-specific information, and our finding that the more information we feed a network the better it performs, suggests that the evolutionary scheme can be applied to a wide variety of experimental systems.

Dynamical trajectories are stochastic, even given a fixed protocol (policy), and so networks that perform well in one generation may be eliminated the next. This can happen if, for example, a certain protocol promotes nucleation, the onset time for which varies from one trajectory to another. By the same token, the best yield can decrease from one generation to the next, and independent trajectories generated using a given protocol have a range of yields. To account for this effect one could place evolutionary requirements on the yield associated with many independent trajectories using the same protocol. Here we opted not to do this, reasoning that over the course of several generations the evolutionary process will naturally identify protocols that perform well when measured over many independent trajectories. We demonstrate this feature in Sec. III, where independent trajectories produced under slow cooling display a wide variety of outcomes, but independent trajectories generated by evolved protocols display relatively well-defined ones.

III. PROMOTING SELF-ASSEMBLY

In Fig. 2 we consider the “3.12.12” disk of Ref. [68], which has three chemically specific patches whose bisectors are separated by angles $\pi/3$ and $5\pi/6$. This disk can form a structure equivalent to the 3.12.12 Archimedean tiling (a tiling with one 3-gon and two 12-gons around each vertex). The number of 12-gons N_{12} counts the number of unit cells of the structure and so is a suitable order parameter for evolutionary search. This structure is a difficult target for self-assembly because its unit cell is large and must form from floppy intermediates, the nature of which gives plenty of scope for mistakes of binding and kinetic trapping. As a result, while intuitive protocols allow assembly to proceed, they do so with relatively low fidelity. In Fig. 2(a) we show the outcome of “cooling” simulations done at three different rates. As for evolutionary simulations, we start from control-parameter values $\epsilon/k_B T = 3$ and $\mu/k_B T = 2$, where the equilibrium state is a sparse gas of largely unassociated disks. Every $t_{\Delta T}$ Monte Carlo steps we increase $\epsilon/k_B T$ by a value 0.075. We carried out 50 independent simulations at each cooling rate. As the rate of cooling decreases, the yield increases, but achieving much more than 50 unit cells of the target material is time-consuming: Single trajectories at each of the

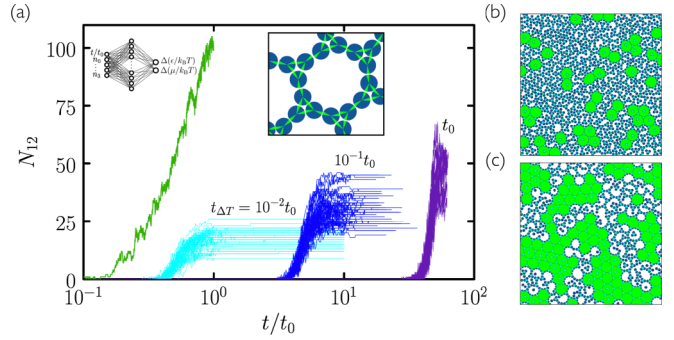


FIG. 2. (a) A 3-patch disk with chemically selective patches can form a structure equivalent to the 3.12.12 Archimedean tiling [68], a tiling with one 3-gon and two 12-gons around each vertex (inset). Slow cooling simulations, in which the disk interaction strength $\epsilon/k_B T$ is increased by 0.075 every $t_{\Delta T}$ Monte Carlo steps, give rise to the numbers of 12-gons N_{12} (the number of unit cells of the desired structure) shown in the plot: We show 50 independent trajectories at each of three cooling rates. Neural networks learn to control $\epsilon/k_B T$ and $\mu/k_B T$ in order to greatly exceed these yields, in a fraction of the time (green line at left). Panels (b) and (c) show snapshots of structures produced by slow cooling and a neural-network protocol, respectively, with 12-gons colored green.

cooling rates take, respectively, of order an hour, a day, and a week of CPU time on a single processor. Clearly, substantial improvement using this protocol would require prohibitively long simulations.

Search using evolutionary learning results in protocols that can greatly exceed the yield of cooling simulations, in a fraction of the time [an example is shown at left in Fig. 2(a)]. In Figs. 3(a)–3(c) we show results obtained using the time network within the evolutionary scheme of Fig. 1. Generation-0 trajectories are controlled by essentially random protocols, and many (e.g., those that involve weakening of interparticle bonds) result in no assembly (see Fig. 4). Some protocols result in low-quality assembly (comparable to that seen in the fastest cooling protocols of Fig. 2), and the best of these are used to create generation 1. Fig. 3(a) shows that assembly gets better with generation number: evolved networks learn to promote assembly of the desired structure. The protocols leading to these structures are shown in Figs. 3(b) and 3(c): Early generation networks tend to strengthen bonds (“cool”) quickly and concentrate the substrate, while later-generation networks strengthen bonds *more* quickly but also promote evacuation of the substrate. This strategy appears to reduce the number of obstacles to the closing of the large and floppy intermediate structures. The most advanced networks further refine these bond-strengthening and substrate-evacuation protocols.

The microscopic network [Figs. 3(d)–3(f)] produces slightly more nuanced versions of the time-network protocols and leads to better assembly. Thus, networks given access to configurational information learn more completely than those that know only the elapsed time of the procedure, even though the information they are given does not directly relate to the quality of assembly. In Fig. 5 we show in more detail a trajectory produced by the best generation-18 microscopic network. The self-assembly dynamics that results is hierarchical assembly of the type seen in Ref. [68], in which trimers

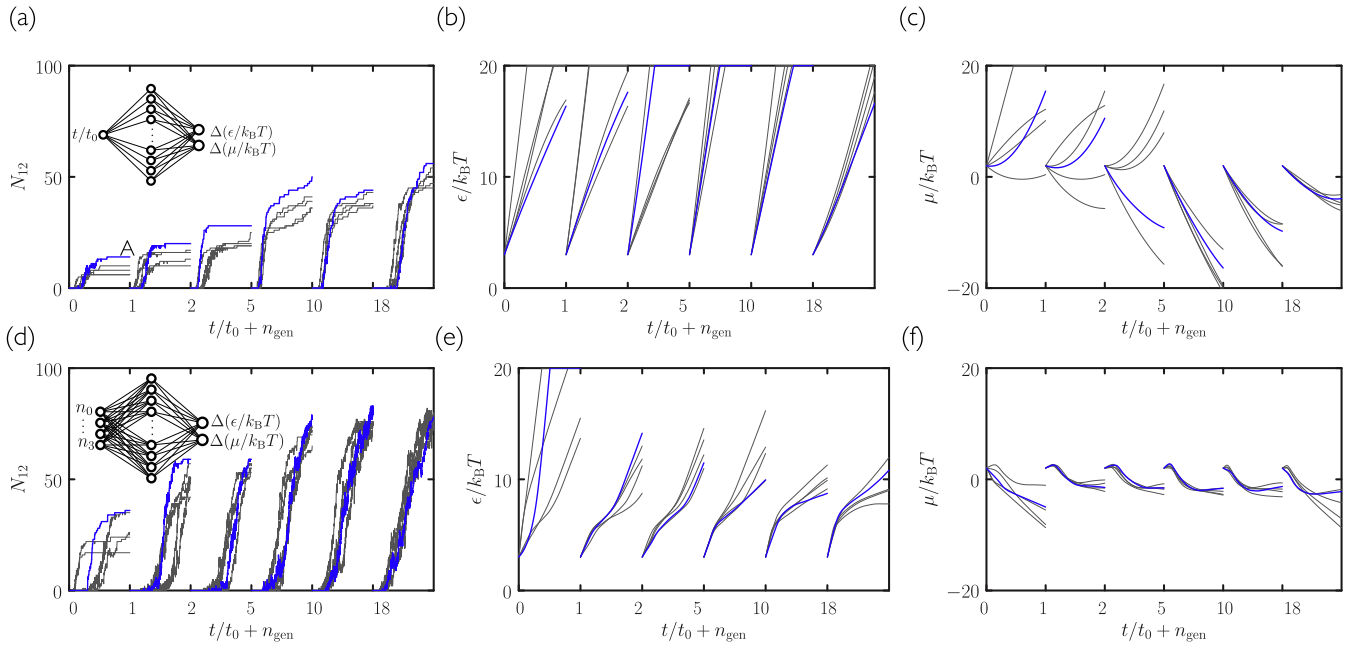


FIG. 3. Evolutionary learning of self-assembly protocols using the 3.12.12 disk of Fig. 2. (a) The time network used with this disk, within the evolutionary scheme of Fig. 1, produces progressively better yields of 12-gons with generation. We show the top 5 yields per generation, with the best shown in blue (dark gray). The protocols leading to these yields are shown in (b) and (c), the better yields corresponding to rapid cooling and evacuation of the substrate. (d) The microscopic network used in the same evolutionary scheme produces better yields than the time network, using (e) and (f) similar but slightly more nuanced protocols. Networks that take both temporal and microscopic information, or two networks used in sequence, produce better yields still: see Fig. 7.

(3-gons) form first and networks of trimers then form 12-gons, but is a more extreme version: In Fig. 5 we see that almost all the 3-gons made by the system form before the 12-gons begin

to form. Thus the network has adopted a two-stage procedure in an attempt to maximize yield.

Networks given either temporal or microscopic information have therefore learned to promote self-assembly, without

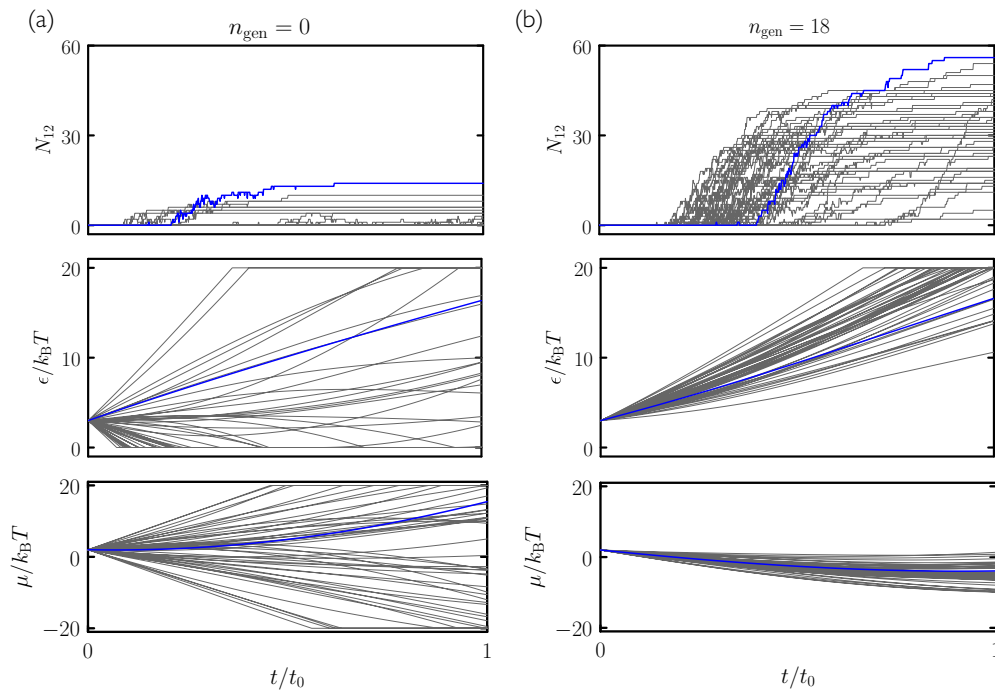


FIG. 4. (a) Generation-0 trajectories of the time network applied to the 3.12.12 disk; most networks fail to produce assembly. (b) Generation-18 trajectories generally result in much better assembly. However, note that some networks, although they are offspring of successful generation-17 networks, result in low-quality assembly.

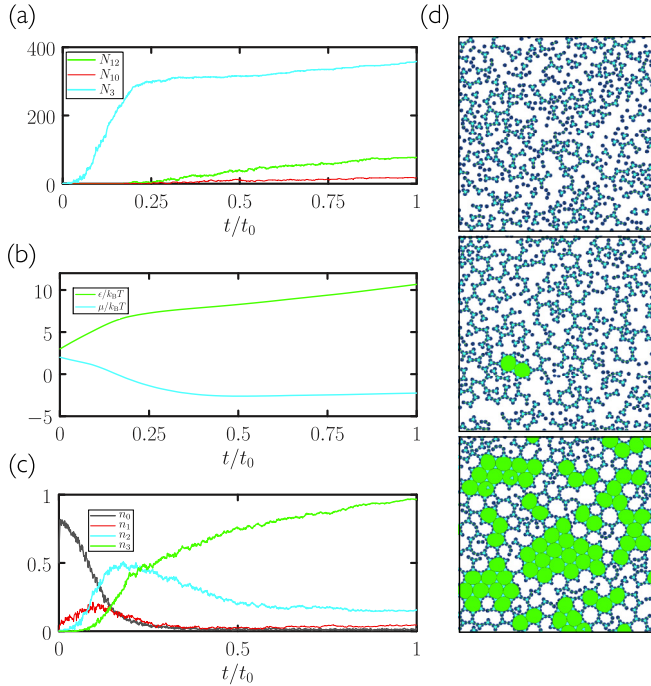


FIG. 5. A self-assembly trajectory produced by the best generation-18 microscopic network of Fig. 3(d)–3(f). Panel (a) and the time-ordered snapshots in (d) show the dynamics to be hierarchical in an extreme way, with most 3-gons (blue) forming before 12-gons (green) are made. Snapshot times are $t/t_0 = 0.17, 0.25, 1$, from top to bottom. Defects, such as disordered regions and 10- and 14-gons, also form. Panel (b) shows the temperature and chemical potential protocols chosen by the network, and (c) shows the inputs to the network.

any external direction beyond an assessment, at the end of the trajectory, of which outcomes were best. Moreover, the quality of assembly considerably exceeds the quality of intuition-driven cooling procedures, and proceeds much more quickly. In Fig. 6 we compare trajectories and assembled structures produced by cooling and by two different networks: The networks produce better structures, even though they are constrained to act over much shorter times. Here we observe the counterintuitive result of rapidly varying nonequilibrium protocols producing better-quality assembly than a slow-cooling procedure designed (at least in an intuitive sense) to promote “near equilibrium” conditions [73].

Yield under the evolutionary protocols can be increased by providing more data to the neural network. In Fig. 7 we show that a neural network provided with both temporal and microscopic information outperforms both the time- and the microscopic networks of Fig. 3. Yield can also be increased by using two neural networks, one after the other, trained independently [see Fig. 6(c) and Figs. 7(e)–7(h)]. In these cases the yield of material reaches more than double that obtained under a slow-cooling protocol. Protocols learned by these neural networks are distinctly different at early and late times, suggestive of distinct growth and annealing stages: After an initial stage of rapid growth under cool and sparse conditions, networks heat the substrate and make it more dense, apparently in order to promote error correction.

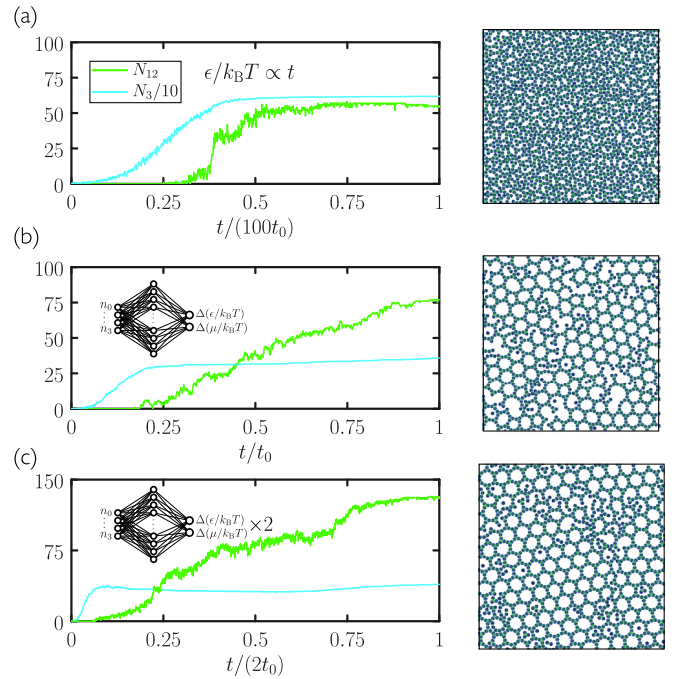


FIG. 6. We compare trajectories produced by (a) the slowest cooling rate shown in Fig. 2, (b) the best generation-18 microscopic network from Fig. 3(d)–3(f), and (c) a generation-12 procedure using two microscopic networks in sequence; see Fig. 7(e)–7(h). The neural networks produce better assembly than the cooling protocol (measured by the 12-gon count, i.e., the number of unit cells of the desired structure), and do so 50 or 100 times faster. The snapshots at right are taken at the end of the three trajectories. In (a), the 3.12.12 structure contains many smaller species in its pores. Note also that some of the larger closed loops in these images are 10-gons or 14-gons; the polygon representation of Fig. 5 picks out 12-gons more clearly.

Here we have provided no prior input to the neural net to indicate what constitutes a good assembly protocol. One could alternatively survey parameter space as thoroughly as possible, using intuition and prior experience, before turning to evolution. In such cases generation-0 assembly would be better than under randomized protocols. However, we found that even when generation-0 assembly was already of high quality, the evolutionary procedure was able to improve it. In Fig. 8 we consider evolutionary learning using the regular three-patch disk without patch-type specificity [56,68]. This disk forms the honeycomb network so readily that the best examples of assembly using 50 randomly chosen protocols (generation 0) are already good. Nonetheless, evolution using the time network or microscopic network is able to improve the quality of assembly, with the microscopic network again performing better.

IV. POLYMORPH SELECTION

Controlling the polymorph into which a set of molecules will self-assemble is a key consideration in industrial procedures such as drug crystallization [21–24]. Here we show that evolutionary search can be used to find protocols able to direct the self-assembly of a set of model molecules into either of

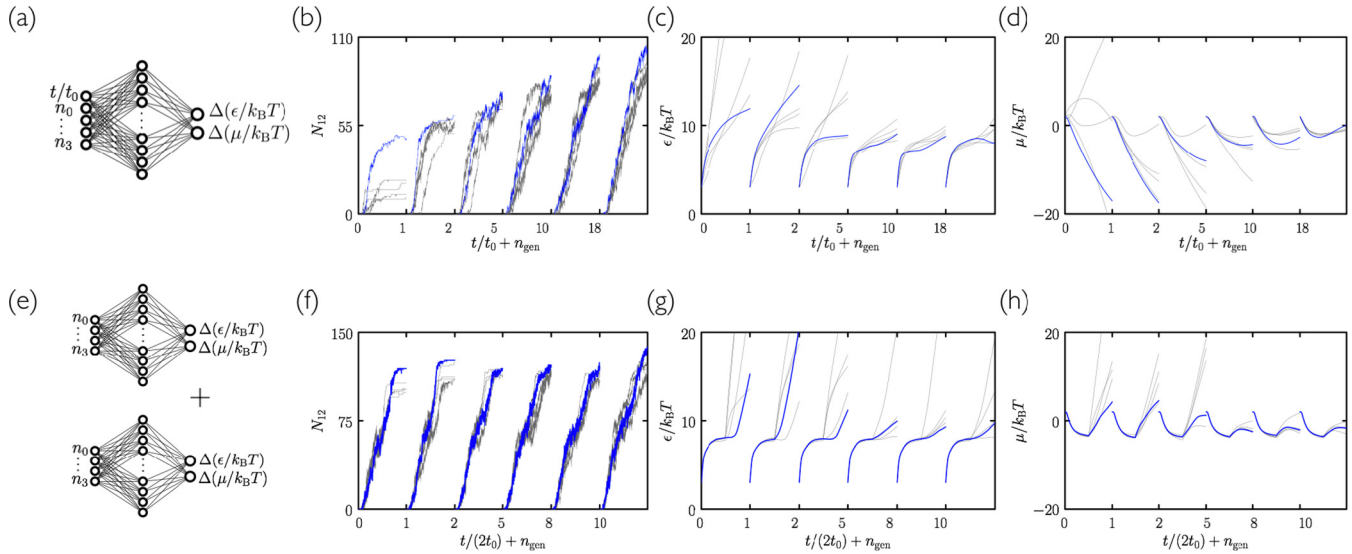


FIG. 7. (a) A neural network that combines temporal and microscopic information outperforms both the time- and microscopic networks of Fig. 3. Panel (b) shows the yield of the top 5 of 50 trajectories for certain generations; panels (c) and (d) show the associated values of $\epsilon/k_B T$ and $\mu/k_B T$, respectively. The protocol learned by this network learned in Fig. 3 but with more pronounced nonmonotonicity: At later times the substrate is heated and made more dense, which appears to facilitate annealing of the structures grown under cold, sparse conditions. A generation-18 trajectory of this network is shown in Fig. 2(a) (left). In panels (e)–(h) we show results using a two-step procedure: The best generation-18 microscopic network from Fig. 3 is applied for time t_0 , and then a second microscopic network is applied and trained for a period t_0 . This procedure identifies a protocol similar to that shown in the upper panels, whose early and late stages are suggestive of distinct growth and annealing conditions. These strategies produce yield of order twice that obtained under slow cooling; see Figs. 2 and 6.

two competing polymorphs. In doing so, the procedure learns strategies that provide physical insight into the system under study.

In Fig. 9 we consider a 4-patch disk with angles $\pi/3$ and $2\pi/3$ between patch bisectors. This disk can form a structure equivalent to the 3.6.3.6 Archimedean tiling (a tiling with two 3-gons and two 6-gons around each vertex), or a rhombic

structure. Particles have equal energy within the bulk of each structure, and at zero pressure (the conditions experienced by a cluster growing in isolation in a substrate) there is no thermodynamic preference for one structure over the other. Independent trajectories generated under slow cooling (gray circles) therefore display nucleation of either or both polymorphs, in an unpredictable way (see also Fig. 12). The

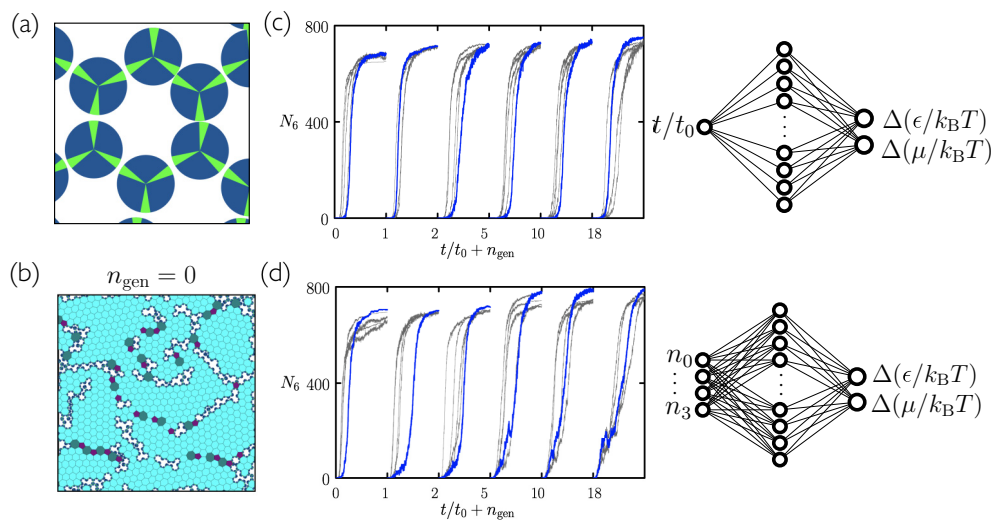


FIG. 8. Evolutionary learning of self-assembly protocols with the regular three-patch disk without patch specificity. This disk forms the honeycomb network (a) so readily that assembly using 50 randomly chosen protocols (generation-0) is already good; see panel (b), in which 6-gons are colored light blue (gray). Nonetheless, evolution using the time network (c) or microscopic network (d) can improve the quality of assembly, and, again, the microscopic network is better than the time one. We show the top five trajectories per generation, with the best shown in blue (dark gray).

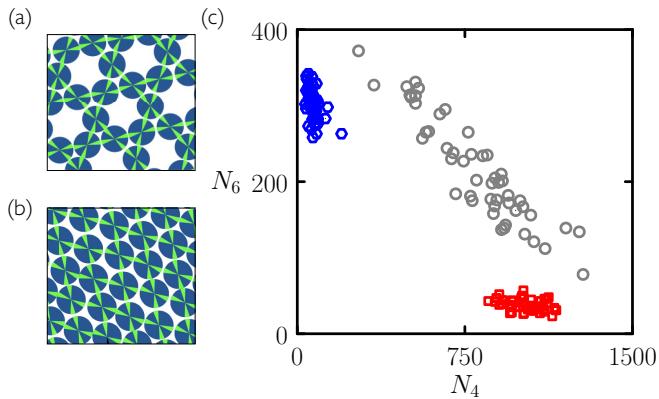


FIG. 9. A 4-patch disk with angles $\pi/3$ and $2\pi/3$ between patch bisectors can form (a) a structure equivalent to the 3.6.3.6 Archimedean tiling or (b) a rhombic structure [68]. The number of 6-gons (N_6) or 4-gons (N_4) serve as order parameters for these structures. (c) Cooling (at the slowest rate shown in Fig. 2) causes nucleation and growth of the two polymorphs on a timescale of order $20t_0$ (see Fig. 12). The outcome of 50 such trajectories consists of either or both polymorphs, in an unpredictable way (gray circles). By contrast, the evolutionary scheme of Fig. 1 can produce neural networks able to select either polymorph with high fidelity. Blue (dark gray) hexagons [respectively, red (light gray) squares] correspond to 50 trajectories, of length t_0 , using a single generation-10 microscopic neural network evolved so as to maximize N_6 (respectively, N_4); see Fig. 10.

3.6.3.6 polymorph can be selected by making the patches chemically selective [68], but here we do not do this. Instead, we show that evolutionary search can be used to develop protocols able to choose between these two polymorphs.

In Fig. 10 we consider evolutionary learning of self-assembly protocols using time- and microscopic networks instructed to promote either the parameter N_4 or the parameter N_6 . In both cases we see steadily increasing counts, with generation, of the required order parameter, with the microscopic network again performing better. The lower two rows show the evolution of the strategies chosen by each network, with time- and microscopic networks learning qualitatively similar protocols for promotion of a given order parameter.

In Fig. 11 we show in more detail one trajectory per strategy obtained using generation-10 microscopic neural networks. Left-hand panels pertain to a trajectory produced by a neural network evolved to promote 6-gons, while right-hand panels pertain to a trajectory produced by a neural network evolved to promote 4-gons. In each case, neural networks have learned to promote one polymorph and so suppress the other. Both examples of assembly display defects and grain boundaries, but the specified polymorphs cover substantial parts of the substrate. In the case considered in Sec. III we already knew how to promote assembly, by cooling—although the evolutionary protocol learned to do it more quickly and with higher fidelity—but here we did not possess advance knowledge of how to do polymorph selection using protocol choice.

In Fig. 11, inspection of the snapshots (a), the polygon counts (b), and the control-parameter histories (c) provide insight into the selection strategies adopted by the networks. To select the 3.6.3.6 tiling (left panels) the network has

induced a tendency for particles to leave the surface (small $\mu/k_B T$) and for bonds to be moderately strong (moderate $\epsilon/k_B T$). The balance of these things appears to be such that trimers (3-gons), in which each particle has two engaged bonds, can form. Trimers serve as a building block for the 3.6.3.6 structure, which then forms hierarchically as the chemical potential is increased (and the bond strength slightly decreased). By contrast, the rhombic structure appears to be unable to grow because it cannot form hierarchically from collections of rhombi (which also contain particles with two engaged bonds): Growing beyond a single rhombus involves the addition of particles via only one engaged bond, and these particles are unstable, at early times, to dissociation.

To select the rhombic structure (right panels) the network selects moderate bond strength and concentrates the substrate by driving $\mu/k_B T$ large. In a dense environment it appears that the rhombic structure is more accessible kinetically than the more open 3.6.3.6 network. In addition, in a dense environment there is a thermodynamic preference for the more compact rhombic polymorph, a factor that may also contribute to selection of the latter. Note that simply causing $\mu/k_B T$ to increase with time is not sufficient to produce the rhombic polymorph in high yield: Early generation networks adopt just such a strategy, but high yield for later generations requires a particular balance of bond strength and chemical potential.

The microscopic network receives information periodically from the system, but the information it receives—the number of particles with certain numbers of engaged bonds—does not distinguish between the bulk forms of the two polymorphs. Networks must therefore learn the relationships between these inputs, their resulting actions, and the final-time order parameter. The time network learns qualitatively similar protocols, albeit with slightly less effectiveness, with no access to the microscopic state of the system.

Returning to Fig. 9(c), we show the results of 50 independent trajectories of length t_0 carried out using a single generation-10 microscopic network evolved to promote 6-gons (blue hexagons), and the results of 50 independent trajectories of length t_0 carried out using a single generation-10 microscopic network evolved to promote 4-gons (red squares). In both cases the networks reliably promote one polymorph and suppress the other, in contrast to slow-cooling simulations whose outcome is unpredictable. In this case the conventional nucleation-and-growth pathway induced by slow cooling provides no control over polymorph selection, while the pathways induced by the neural networks—one of which is strongly hierarchical—do. In addition, as in Sec. III, assembly under the network protocols is much faster than under slow cooling; see Fig. 12.

V. CONCLUSIONS

We have shown that a self-assembly kinetic yield net trained by evolutionary reinforcement learning [43–50] can control self-assembly protocols in molecular simulations. Networks learn to promote the assembly of desired structures, or choose between polymorphs. In the first case, networks reproduce the structures produced by previously known protocols, but faster and with higher fidelity; in the second case they identify strategies previously unknown, and from which we

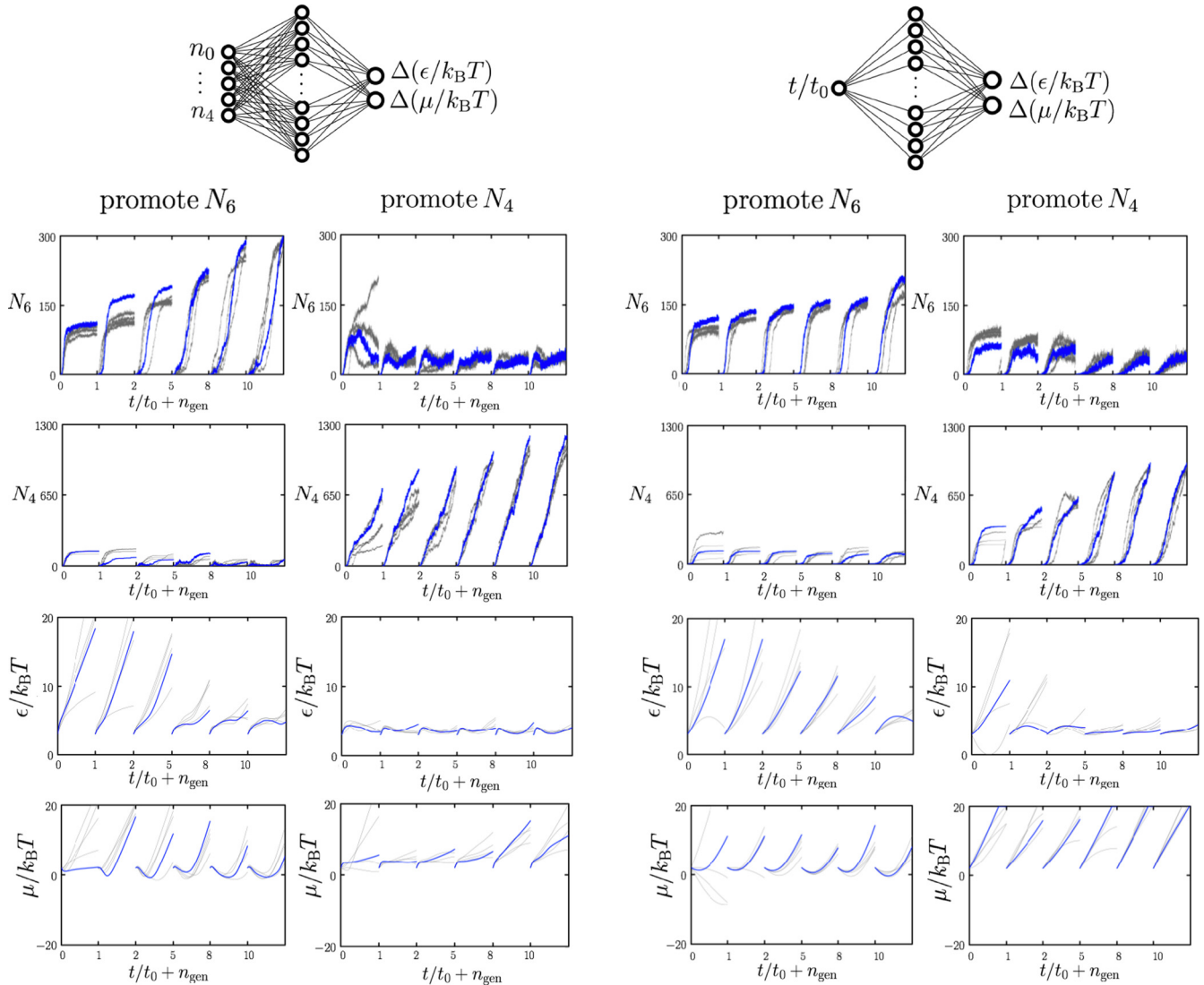


FIG. 10. Evolutionary learning of self-assembly protocols for the 4-patch disk of Fig. 9 using the microscopic network (left two columns) or the time network (right two columns). Networks instructed to maximize the number of 6-gons (columns 1 and 3) or 4-gons (columns 2 and 4) learn to promote the assembly of the 3.6.3.6 tiling or the rhombic structure. As in the other cases studied, the microscopic network is more effective than the time network. We show the top five protocols per generation, with the best shown in blue (dark gray).

can extract physical insight. Networks that take as input only the elapsed time of the protocol are effective, and networks that take as input microscopic information from the system are more so. This comparison indicates that this scheme can be applied to a wide range of experiments, regardless of how much microscopic information is available as assembly proceeds.

The problem we have addressed falls in the category of reinforcement learning in the sense that the neural network learns to perform actions (choosing new values of the control parameters) given an observation. The evolutionary approach we have applied to this problem requires only the assessment of a desired order parameter (here the polygon count N_α) at the end of a trajectory. This is an important feature because in self-assembly the best-performing trajectories at short times are not necessarily the best-performing trajectories at the desired observation time; see, e.g., Fig. 4. Self-assembly is inherently a “sparse-reward” problem. For this reason it is

not obvious that value-based reinforcement-learning methods [25] are ideally suited to a problem such as self-assembly: Rewarding “good” configurations at early times may not result in favorable outcomes at later times. This is only speculation on our part, however; which of the many ways of doing reinforcement learning is best for self-assembly is an open question.

Our results demonstrate proof of principle, and can be extended or adapted in several ways. We allow networks to act 1000 times per trajectory, in order to mimic a system in which we have only occasional control; the influence of a network could be increased by allowing it to act more frequently. We have chosen the hyperparameters of our scheme (mutation step size, neural network width, network activation functions, number of trajectories per generation) using values that seemed reasonable and that we subsequently observed to work, but these could be optimized (potentially by evolutionary search).

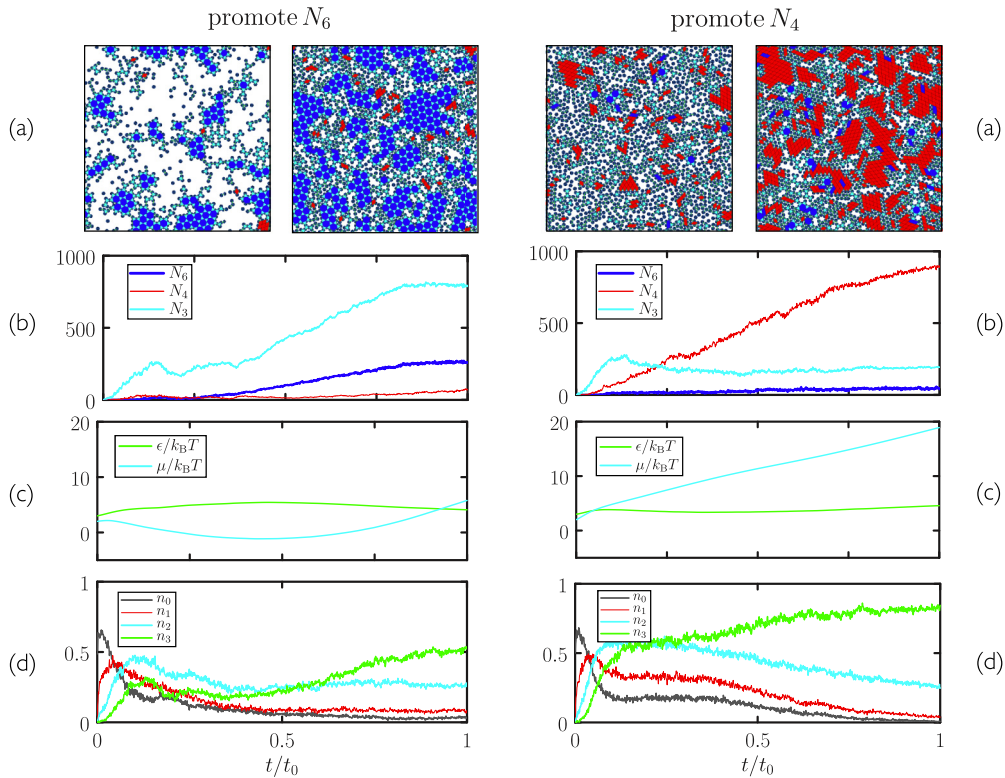


FIG. 11. Generation-10 trajectories of the microscopic network from Fig. 10, evolved to promote either 6-gons (left column) or 4-gons (right column). We show (a) time-ordered snapshots, (b) polygon counts, (c) neural network outputs, and (d) neural network inputs. In snapshots, 6-gons are dark blue (dark gray), 3-gons are light blue (light gray), and 4-gons are red (lighter gray).

We end by noting that the scheme we have used is simple to implement. The network architectures we have used are standard and can be straightforwardly adapted to handle an arbitrary number of inputs (system data) and outputs (changes

of system control parameters). The mutation protocol is simple to implement. In addition, we have shown that learning can be effective using a modest number of trajectories (50) per generation. The evolutionary scheme should therefore be

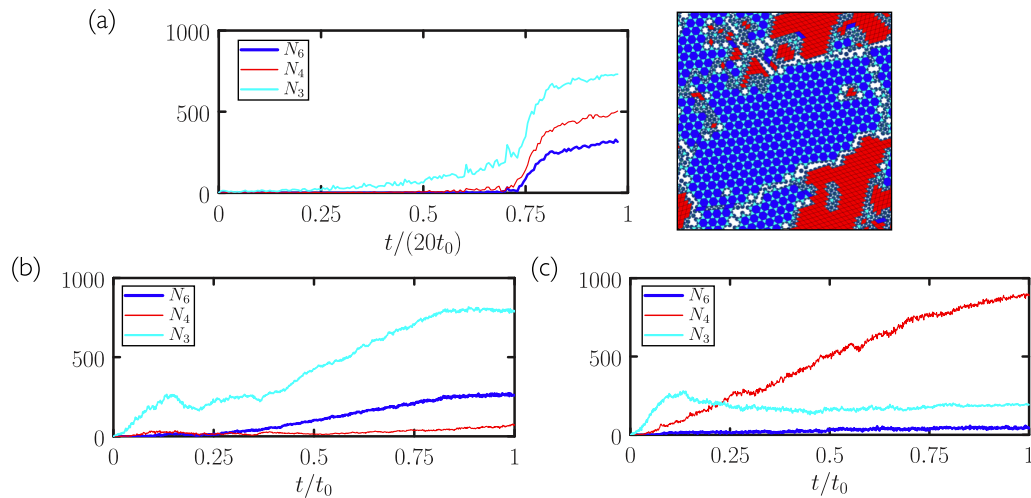


FIG. 12. (a) Slow cooling of the 4-patch disk of Fig. 9 results in nucleation and growth of either polymorph type, in unpredictable quantities; see the gray circles in Fig. 9(c). Here both polymorphs appear. In the snapshot, 6-gons are dark blue (dark gray), 3-gons are light blue (light gray), and 4-gons are red (lighter gray). Note that the unit cells of these polymorphs are smaller and more mechanically rigid than that of the 3.12.12 structure of Fig. 2, and so the 3.6.3.6 and rhombic polymorphs assemble better under slow cooling than does the 3.12.12 structure. However, there exists no mechanism during slow nucleation and growth to reliably select one polymorph over the other [(b) and (c)]. By contrast, the non-nucleation mechanisms generated by neural networks evolved to promote 6-gons (b) or 4-gons (c) result in more predictable outcomes; see the blue (dark gray) and red (light gray) symbols in Fig. 9(c).

applicable to a broad range of experimental or computational systems. The results shown here have been achieved with no human input beyond the specification of which order parameter to promote, pointing the way to the design of synthesis protocols by artificial intelligence.

ACKNOWLEDGMENTS

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. I.T. performed work at the National Research Council of Canada under the auspices of the AI4D and MCF Programs.

APPENDIX: NEURAL NETWORK

Each network is a fully connected architecture with n_i input nodes, $n_h = 1000$ hidden nodes, and $n_o = 2$ output nodes. Let the indices $i \in \{0, \dots, n_i - 1\}$, $j \in \{1, \dots, n_h\}$, and $k \in \{1, \dots, n_o\}$ label nodes in the input, hidden, and output layers,

respectively. Let $w_{\alpha\beta}$ be the weight connecting nodes α and β , and let b_j be the bias applied to hidden-layer node j . Then the two output nodes take the values

$$S_k = n_h^{-1} \sum_j S_j w_{jk}, \quad (\text{A1})$$

where

$$S_j = \tanh \left(\sum_i S_i w_{ij} + b_j \right), \quad (\text{A2})$$

and S_i denotes the input-node value(s). For the time network we have $n_i = 1$ and $S_0 = t/t_0$. For the microscopic network we have $n_i = P + 1$, where P is the number of patches on the disk, and S_i is the number of particles in the simulation box having $i \in \{0, 1, \dots, P\}$ engaged patches (divided by 1000). The mixed time-microscopic network of Fig. 7 uses both t/t_0 and the S_i as inputs. The output-node values are taken to be the changes $\Delta(\mu/k_B T)$ and $\Delta(\epsilon/k_B T)$, provided that $\mu/k_B T$ and $\epsilon/k_B T$ remain in the intervals $[-20, 20]$ and $[0, 20]$, respectively.

-
- [1] G. M. Whitesides, J. P. Mathias, and C. T. Seto, Molecular self-assembly and nanochemistry: A chemical strategy for the synthesis of nanostructures, *Science* **254**, 1312 (1991).
- [2] P. L. Biancaniello, A. J. Kim, and J. C. Crocker, Colloidal Interactions and Self-Assembly Using DNA Hybridization, *Phys. Rev. Lett.* **94**, 058302 (2005).
- [3] S. Y. Park, A. K. R. Lytton-Jean, B. Lee, S. Weigand, G. C. Schatz, and C. A. Mirkin, DNA-programmable nanoparticle crystallization, *Nature* **451**, 553 (2008).
- [4] D. Nykypanchuk, M. M. Maye, D. van der Lelie, and O. Gang, DNA-guided crystallization of colloidal nanoparticles, *Nature* **451**, 549 (2008).
- [5] Y. Ke, L. L. Ong, W. M. Shih, and P. Yin, Three-dimensional structures self-assembled from dna bricks, *Science* **338**, 1177 (2012).
- [6] W. Pfeifer and B. Saccà, Synthetic dna filaments: From design to applications, *Biol. Chem.* **399**, 773 (2018).
- [7] J. P. K. Doye, A. A. Louis, and M. Vendruscolo, Inhibition of protein crystallization by evolutionary negative design, *Phys. Biol.* **1**, P9 (2004).
- [8] F. Romano and F. Sciortino, Colloidal self-assembly: Patchy from the bottom up, *Nat. Mater.* **10**, 171 (2011).
- [9] S. C. Glotzer, M. J. Solomon, and N. A. Kotov, Self-assembly: From nanoscale to microscale colloids, *AIChE J.* **50**, 2978 (2004).
- [10] J. P. K. Doye, A. A. Louis, I. C. Lin, L. R. Allen, E. G. Noya, A. W. Wilber, H. C. Kok, and R. Lyus, Controlling crystallization and its absence: Proteins, colloids and patchy models, *Phys. Chem. Chem. Phys.* **9**, 2197 (2007).
- [11] D. C. Rapaport, Modeling capsid self-assembly: Design and analysis, *Phys. Biol.* **7**, 045001 (2010).
- [12] A. Reinhardt and D. Frenkel, Numerical Evidence for Nucleated Self-Assembly of DNA Brick Structures, *Phys. Rev. Lett.* **112**, 238103 (2014).
- [13] J. J. De Yoreo, P. U. P. A. Gilbert, N. A. J. M. Sommerdijk, R. L. Penn, S. Whitelam, D. Joester, H. Zhang, J. D. Rimer, A. Navrotsky, J. F. Banfield *et al.*, Crystallization by particle attachment in synthetic, biogenic, and geologic environments, *Science* **349**, aaa6760 (2015).
- [14] A. Murugan, J. Zou, and M. P. Brenner, Undesired usage and the robust self-assembly of heterogeneous structures, *Nat. Commun.* **6**, 6203 (2015).
- [15] S. Whitelam and R. L. Jack, The statistical mechanics of dynamic pathways to self-assembly, *Annu. Rev. Phys. Chem.* **66**, 143 (2015).
- [16] M. Nguyen and S. Vaikuntanathan, Design principles for nonequilibrium self-assembly, *Proc. Natl. Acad. Sci. USA* **113**, 14231 (2016).
- [17] S. Whitelam, L. O. Hedges, and J. D. Schmit, Self-Assembly at a Nonequilibrium Critical Point, *Phys. Rev. Lett.* **112**, 155504 (2014).
- [18] R. B. Jadrich, B. A. Lindquist, and T. M. Truskett, Probabilistic inverse design for self-assembling materials, *J. Chem. Phys.* **146**, 184103 (2017).
- [19] J. F. Lutsko, How crystals form: A theory of nucleation pathways, *Sci. Adv.* **5**, eaav7399 (2019).
- [20] Z. Fan and M. Grunwald, Orientational order in self-assembled nanocrystal superlattices, *J. Am. Chem. Soc.* **141**, 1980 (2019).
- [21] J. Chen, B. Sarma, J. M. B. Evans, and A. S. Myerson, Pharmaceutical crystallization, *Cryst. Growth Des.* **11**, 887 (2011).
- [22] T. Threlfall, Crystallisation of polymorphs: Thermodynamic insight into the role of solvent, *Org. Process Res. Dev.* **4**, 384 (2000).
- [23] N. Rodríguez-hornedo and D. Murphy, Significance of controlling crystallization mechanisms and kinetics in pharmaceutical systems, *J. Pharm. Sci.* **88**, 651 (1999).

- [24] B. Y. Shekunov and P. York, Crystallization processes in pharmaceutical technology and drug delivery design, *J. Cryst. Growth* **211**, 122 (2000).
- [25] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction* (MIT Press, Cambridge, MA, 2018).
- [26] C. J. C. H. Watkins and P. Dayan, Q-learning, *Mach. Learn.* **8**, 279 (1992).
- [27] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, Playing atari with deep reinforcement learning, *NIPS Deep Learning Workshop* (Lake Tahoe, 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.*, Human-level control through deep reinforcement learning, *Nature* **518**, 529 (2015).
- [29] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, The arcade learning environment: An evaluation platform for general agents, *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, Asynchronous methods for deep reinforcement learning, in *Proceedings of the 33rd International Conference on Machine Learning, New York, NY (JMLR: W&CP, 2016)*, Vol. 48, pp. 1928–1937.
- [31] Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. de Las Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq *et al.*, Deepmind control suite, [arXiv:1801.00690](https://arxiv.org/abs/1801.00690) (2018).
- [32] E. Todorov, T. Erez, and Y. Tassa, Mujoco: A physics engine for model-based control, in *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'13)* (IEEE, Los Alamitos, CA, 2012), pp. 5026–5033.
- [33] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (John Wiley & Sons, New York, 2014).
- [34] A. Asperti, D. Cortesi, and F. Sovrano, Crawling in Rogue's dungeons with (Partitioned) A_3C , in *International Conference on Machine Learning, Optimization, and Data Science, LOD 2018, Lecture Notes in Computer Science*, Vol. 11331 (Springer, Cham, 2018), pp. 264–275.
- [35] M. Riedmiller, Neural fitted q iteration—first experiences with a data efficient neural reinforcement learning method, in *Proceedings of the European Conference on Machine Learning* (Springer, Berlin, 2005), pp. 317–328.
- [36] M. Riedmiller, T. Gabel, R. Hafner, and S. Lange, Reinforcement learning for robot soccer, *Auton. Robots* **27**, 55 (2009).
- [37] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, Proximal policy optimization algorithms, [arXiv:1707.06347](https://arxiv.org/abs/1707.06347) (2017).
- [38] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, Openai gym, [arXiv:1606.01540](https://arxiv.org/abs/1606.01540) (2016).
- [39] M. Kempka, M. Wydmuch, G. Runc, J. Toczec, and W. Jaśkowski, Vizdoom: A doom-based ai research platform for visual reinforcement learning, in *Proceedings of the 2016 IEEE Conference on Computational Intelligence and Games (CIG'16)* (IEEE, Los Alamitos, CA, 2016), pp. 1–8.
- [40] M. Wydmuch, M. Kempka, and W. Jaśkowski, Vizdoom competitions: Playing doom from pixels, [arXiv:1809.03470](https://arxiv.org/abs/1809.03470) (2018).
- [41] 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.*, Mastering the game of go with deep neural networks and tree search, *Nature* **529**, 484 (2016).
- [42] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, Mastering the game of go without human knowledge, *Nature* **550**, 354 (2017).
- [43] F. P. Such, V. Madhavan, E. Conti, J. Lehman, K. O. Stanley, and J. Clune, Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning, [arXiv:1712.06567](https://arxiv.org/abs/1712.06567) (2017).
- [44] J. H. Holland, Genetic algorithms, *Sci. Am.* **267**, 66 (1992).
- [45] D. B. Fogel and L. C. Stayton, On the effectiveness of crossover in simulated evolutionary optimization, *BioSystems* **32**, 171 (1994).
- [46] J. Lehman, J. Chen, J. Clune, and K. O. Stanley, Es is more than just a traditional finite-difference approximator, in *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '18, 15–19 July 2018, Kyoto, Japan* (ACM, 2018), pp. 450–457.
- [47] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever, Evolution strategies as a scalable alternative to reinforcement learning, [arXiv:1703.03864](https://arxiv.org/abs/1703.03864) (2017).
- [48] X. Zhang, J. Clune, and K. O. Stanley, On the relationship between the openai evolution strategy and stochastic gradient descent, [arXiv:1712.06564](https://arxiv.org/abs/1712.06564) (2017).
- [49] J. Lehman, J. Chen, J. Clune, and K. O. Stanley, Safe mutations for deep and recurrent neural networks through output gradients, in *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '18, 15–19 July 2018, Kyoto, Japan* (ACM, 2018), pp. 117–124.
- [50] E. Conti, V. Madhavan, F. P. Such, J. Lehman, K. Stanley, and J. Clune, Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents, in *Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada* (2018), pp. 5027–5038.
- [51] Z. Zhang and S. C. Glotzer, Self-assembly of patchy particles, *Nano Lett.* **4**, 1407 (2004).
- [52] F. Romano, E. Sanz, and F. Sciortino, Crystallization of tetrahedral patchy particles in silico, *J. Chem. Phys.* **134**, 174502 (2011).
- [53] F. Sciortino, E. Bianchi, J. F. Douglas, and P. Tartaglia, Self-assembly of patchy particles into polymer chains: A parameter-free comparison between wertheim theory and monte carlo simulation, *J. Chem. Phys.* **126**, 194903 (2007).
- [54] E. Bianchi, P. Tartaglia, E. Zaccarelli, and F. Sciortino, Theoretical and numerical study of the phase diagram of patchy colloids: Ordered and disordered patch arrangements, *J. Chem. Phys.* **128**, 144504 (2008).
- [55] G. Doppelbauer, E. Bianchi, and G. Kahl, Self-assembly scenarios of patchy colloidal particles in two dimensions, *J. Phys.: Condens. Matter* **22**, 104105 (2010).
- [56] S. Whitelam, I. Tamblyn, T. K. Haxton, M. B. Wieland, N. R. Champness, J. P. Garrahan, and P. H. Beton, Common Physical Framework Explains Phase Behavior and Dynamics of Atomic, Molecular, and Polymeric Network Formers, *Phys. Rev. X* **4**, 011044 (2014).

- [57] É. Duguet, C. Hubert, C. Chomette, A. Perro, and S. Ravaine, Patchy colloidal particles for programmed self-assembly, *C. R. Chim.* **19**, 173 (2016).
- [58] D. Frenkel and B. Smit, *Understanding Molecular Simulation: From Algorithms to Applications* (Academic Press, Orlando, FL, 1996).
- [59] D. Klotsa and R. L. Jack, Controlling crystal self-assembly using a real-time feedback scheme, *J. Chem. Phys.* **138**, 094502 (2013).
- [60] X. Tang, B. Rupp, Y. Yang, T. D. Edwards, M. A. Grover, and M. A. Bevan, Optimal feedback controlled assembly of perfect crystals, *ACS Nano* **10**, 6791 (2016).
- [61] M. Z. Miskin, G. Khaira, J. J. de Pablo, and H. M. Jaeger, Turning statistical physics models into materials design engines, *Proc. Natl. Acad. Sci. U.S.A.* **113**, 34 (2016).
- [62] A. W. Long, J. Zhang, S. Granick, and A. L. Ferguson, Machine learning assembly landscapes from particle tracking data, *Soft Matter* **11**, 8141 (2015).
- [63] A. W. Long and A. L. Ferguson, Nonlinear machine learning of patchy colloid self-assembly pathways and mechanisms, *J. Phys. Chem. B* **118**, 4228 (2014).
- [64] B. A. Lindquist, R. B. Jadrich, and T. M. Truskett, Communication: Inverse design for self-assembly via on-the-fly optimization, *J. Chem. Phys.* **145**, 111101 (2016).
- [65] B. A. Thurston and A. L. Ferguson, Machine learning and molecular design of self-assembling-conjugated oligopeptides, *Mol. Simul.* **44**, 930 (2018).
- [66] A. L. Ferguson, Machine learning and data science in soft materials engineering, *J. Phys.: Condens. Matter* **30**, 043002 (2017).
- [67] N. Kern and D. Frenkel, Fluid–fluid coexistence in colloidal systems with short-ranged strongly directional attraction, *J. Chem. Phys.* **118**, 9882 (2003).
- [68] S. Whitelam, Minimal Positive Design for Self-Assembly of the Archimedean Tilings, *Phys. Rev. Lett.* **117**, 228003 (2016).
- [69] S. Whitelam, E. H. Feng, M. F. Hagan, and P. L. Geissler, The role of collective motion in examples of coarsening and self-assembly, *Soft Matter* **5**, 1251 (2009).
- [70] L. O. Hedges, <http://vmmc.xyz>.
- [71] T. K. Haxton, L. O. Hedges, and S. Whitelam, Crystallization and arrest mechanisms of model colloids, *Soft Matter* **11**, 9307 (2015).
- [72] The natural way to measure “real” time in such a system is to advance the clock by an amount $(1 + M)^{-1}$ on making an attempted move. Dense systems and sparse systems then take very different amounts of CPU time to run. In order to move simulation generations efficiently through our computer cluster we instead updated the clock by one unit on making a move. In this way we work in the constant event-number ensemble.
- [73] S. Whitelam, Strong bonds and far-from-equilibrium conditions minimize errors in lattice-gas growth, *J. Chem. Phys.* **149**, 104902 (2018).