Turning statistical physics models into materials design engines

Marc Z. Miskin[a,1], Gurdaman Khaira[b], Juan J. de Pablo[b,c], and Heinrich M. Jaeger[a]

[a]James Franck Institute and Department of Physics, The University of Chicago, Chicago, IL 60637; [b]Institute for Molecular Engineering, The University of Chicago, Chicago, IL 60637; and [c]Institute for Molecular Engineering, Argonne National Laboratory, Lemont, IL 60439

Edited by Sharon C. Glotzer, University of Michigan, Ann Arbor, MI, and approved November 3, 2015 (received for review May 13, 2015)

Despite the success statistical physics has enjoyed at predicting the properties of materials for given parameters, the inverse problem, identifying which material parameters produce given, desired properties, is only beginning to be addressed. Recently, several methods have emerged across disciplines that draw upon optimization and simulation to create computer programs that tailor material responses to specified behaviors. However, so far the methods developed either involve black-box techniques, in which the optimizer operates without explicit knowledge of the material's configuration space, or require carefully tuned algorithms with applicability limited to a narrow subclass of materials. Here we introduce a formalism that can generate optimizers automatically by extending statistical mechanics into the realm of design. The strength of this approach lies in its capability to transform statistical models that describe materials into optimizers to tailor them. By comparing against standard black-box optimization methods, we demonstrate how optimizers generated by this formalism can be faster and more effective, while remaining straightforward to implement. The scope of our approach includes possibilities for solving a variety of complex optimization and design problems concerning materials both in and out of equilibrium.

materials design | directed self-assembly | optimization | inverse design

Computer programs that can design material properties have led to exciting, new directions for materials science (1–3). Computational methods have been used to predict crystal (4) and protein (5, 6) structures, yielding the toughest crystals known to mankind (4) and de novo protein configurations unseen in nature (5). Applied to polymers, Monte Carlo methods (7–9) and evolutionary algorithms (10, 11) have paved the way toward optimizing directed self-assembly. Similar methods have been used to identify the crystal structures of patchy, colloidal particles (12). For far-from-equilibrium systems like jammed, metastable aggregates of particles (3), simulation-based optimization has been successfully used to design bulk properties like stiffness (13) and packing density (14) by way of tuning complicated microscale features like particle shape.

However, despite these successes, most of the existing methods work only for narrowly defined classes of materials: Optimization techniques that prove successful at designing one class of materials may struggle or fail on other systems. Thus, designing new materials can require a large investment in trial and error at the level of the algorithm itself, even if, for given parameters, the material's behavior can be simulated easily.

In black-box approaches, the algorithm tunes the material by adjusting control parameters without considering the likelihood of finding the material in microscale configurations. Instead, the optimizer operates in some auxiliary space, defined outside the physical model, and remains ignorant of the statistics in the physical configuration space. On the other hand, for the overwhelming majority of materials, an accurate description of macroscale behavior comes about by explicitly considering the probability of finding the system in certain microscopic configurations. Several materials optimization approaches exist that take this statistical nature into account, examples include optimizers that design spin configurations (15, 16), patchy colloidal particles (17), and self-assembly driven by short-ranged interactions (18). These approaches build heavily upon the specific model defining the material. As a consequence, it is difficult to extend them beyond the material class they are designed for. Evidently, microscale configurations present key statistical information about a material, which is completely ignored by black-box approaches, yet there is no formalism that generically incorporates this information into materials design.

The question we ask is whether microstate information not only can be used to enhance an optimizer's speed and range of applicability, but also can become the cornerstone of an approach that automatically transforms a design goal and a statistical model into an optimizer. One can then construct optimizers that can work on material classes that are as broad as those described by statistical mechanics, without the need for ad hoc modification.

Here we take some first steps toward such a framework. We present a formalism that can be used to transform the capacity to predict material behavior into an optimizer that tunes it. Furthermore we find that our formalism often solves optimization problems faster and more reliably than approaches built around black-box methods.

Deriving the Optimization Equations

Our approach starts by assuming a given model $\rho(x|\lambda_i)$ that predicts the probability of finding a material in some configuration, x. The model depends on the adjustable parameters λ_i and by tuning λ_i, a user can impact the emergent, bulk properties, averaged over configuration space. Thus, design proceeds by tweaking λ_i to promote a desired, user-specified response.

We work toward this goal, starting from the heuristic equation

$$\dot{\rho}(x|\lambda_i) = \rho(x|\lambda_i)[f(x) - \langle f(x)\rangle], \qquad [1]$$

Significance

A fundamental tenet of science is that the properties of a material are intimately linked to the nature of the constituent components. Although there are powerful methods to predict such properties for given components, a key challenge for materials design is the inverse process: identifying the required components and their structural configuration for given target properties. This paper presents a new approach to this challenge. A formalism is introduced that generates algorithms for materials design both under equilibrium and under nonequilibrium conditions and operates without the need for user input beyond a design goal. This formalism is broadly applicable, fast, and robust, and it provides a powerful tool for materials optimization as well as discovery.

Author contributions: M.Z.M., J.J.d.P., and H.M.J. designed research; M.Z.M. and G.K. performed research; M.Z.M. analyzed data; and M.Z.M., G.K., J.J.d.P., and H.M.J. wrote the paper.

The authors declare no conflict of interest.

This article is a PNAS Direct Submission.

[1]To whom correspondence should be addressed. Email: m.miskin@gmail.com.

This article contains supporting information online at www.pnas.org/lookup/suppl/doi:10.1073/pnas.1509316112/-/DCSupplemental.

where the angle brackets denote an average over configurations weighted by ρ, the overdot denotes a derivative with respect to an artificial time that indexes the optimization steps, and $f(x)$ is a function that quantifies how well configuration x represents the user-specified goal. In this way, Eq. **1** attempts to increase the probability of finding the system in states with better than average values of $f(x)$: If a configuration x has a value of $f(x)$ greater than the average, then the probability of finding the system near x increases, for average $f(x)$ it does not change, and for worse than average it decreases.

As written, Eq. **1** assumes that $\rho(x|\lambda_i)$ can be independently set for every possible point in the configuration space, despite the fact that $\rho(x|\lambda_i)$ is constrained by the physics behind the material of interest. In actuality, $\rho(x|\lambda_i)$ can offer only a limited flexibility through the parameters λ_i. Thus, for many problems it will not be possible to exactly satisfy Eq. **1**, although it is possible to make a best approximation to Eq. **1**, given a particular physical distribution. We achieve this by setting the changes to λ_i such that they minimize the average error between the updates implied by Eq. **1** and the actual changes to $\rho(x|\lambda_i)$. Explicitly, we rewrite Eq. **1** as $\dot{\lambda}_i \partial_{\lambda_i} \log[\rho] = f(x) - \langle f \rangle$ and select the $\dot{\lambda}_i$ that minimize the average value of the squared error, $\epsilon = \langle (\dot{\lambda}_i \partial_{\lambda_i} \log[\rho] - [f(x) - \langle f \rangle])^2 \rangle$. The equations of motion for λ_i that minimize ε at a given instant in time are found by setting the partial derivatives with respect to $\dot{\lambda}_i$ equal to zero (*Supporting Information*):

$$\dot{\lambda}_i(t) = \langle \partial_{\lambda_i} \log(\rho) \partial_{\lambda_j} \log(\rho) \rangle^{-1} \langle [f(x) - \langle f(x) \rangle] \partial_{\lambda_j} \log(\rho) \rangle. \qquad [2]$$

Eq. **2** is now a closed expression for λ_i that depends only on expectation values. Thus, it can function as an algorithm: one can draw samples from $\rho(x|\lambda_i)$, use the samples to evaluate the right-hand side of Eq. **2**, and then integrate the equations of motion to generate new, improved parameter settings.

Eq. **2**, and its motivating Eq. **1**, overlap with a surprising number of different fields. For example, the matrix elements in Eq. **2** resemble kinetic coefficients, suggesting the interpretation that $f(x)$ generates a thermodynamic force that pushes the system to solve the design goal (19). Alternatively, Eq. **2** appears in the optimization and mathematics literature as natural gradient descent: It bears the interpretation of a gradient descent method that takes the steepest step such that the change in entropy stays bounded (20–22). Indeed, the matrix in front of Eq. **2** is the Fisher information metric and is constraining the driving force to move in directions of small entropy change (21). This interpretation is also associated with state-of-the-art optimizers like the covariance matrix adaptation evolution strategy (CMA-ES) (22–24); however, here the design parameters λ_i are treated as random variables drawn from a Gaussian distribution irrespective of the design problem, and a version of Eq. **2** is used to update the mean and covariance of this auxiliary distribution. This is in contrast to our proposition that λ_i should be treated as deterministic variables that evolve according to Eq. **2** with randomness entering only at the level of material configurations. If the task considered is changed to finding the best-fit model parameters to a given set of data, then Eq. **2** represents the direction in parameter space that decreases the fit error most efficiently per unit of behavioral change in the model (25). In this scenario, one can consider regularizing Eq. **2** to produce the Levenburg–Marquart algorithm modified to account for the geometric aspects of the optimization. Finally, one can note that the motivating equation, Eq. **1**, is the replicator equation from game theory and evolutionary biology (26–28). Thus, one could also interpret the dynamics as a process of reproduction and competition in a continuous parameter space (26, 27), projected onto $\rho(x|\lambda)$.

Whatever the picture, Eq. **2** has a number of powerful properties. In particular, Eq. **2** is invariant to any invertible reparameterization of λ_i, including rotations, dilations, and translations in parameter space. If the reward function, $f(x)$, is chosen correctly, the velocity flow is also invariant to rank preserving changes in the design goal (22). Thus, there will be no difference in performance between two design problems that differ by coordinate choice over λ_i and/or a rank preserving change in the design goal [e.g., $g(x)$ vs. $\exp(g(x))$], provided the initial values of λ_i are the same. These invariances also provide stability to the algorithm: by making the search algorithm invariant to both the goal function magnitude and the parameterization, the effect of sampling errors gets bounded in a parameterization invariant way. Thus, errors from sampling parameters in Eq. **2** will not cascade, even if the matrix in Eq. **2** becomes ill-conditioned. Altogether, these features greatly simplify the optimization task because, now, the designer is free from worrying about trivial choices surrounding λ_i and the goal function. Details on these points can be found in *Supporting Information*.

For the task of optimizing materials, we stress one further property: By using Eq. **2** optimization takes place in configuration space, rather than in an auxiliary space introduced to define an optimizer. As we will show, this gives a unique advantage in applying Eq. **2** to materials design: More information is used by the optimizer when updating parameters, without incurring an increase in computational cost. Perhaps best of all, this optimizer is constructed by straightforwardly applying the formalism encoded in Eq. **2** to the relevant statistical model. For example, when $\rho(x|\lambda_i)$ is given by the canonical ensemble, $\rho(x|\lambda_i) \propto \exp[-\lambda_i h_i(x)]$, the optimizer follows immediately from Eq. **2** as $\dot{\lambda}_i = -\mathrm{Cov}[h_i(x), h_j(x)]^{-1} \mathrm{Cov}[h_j(x), f(x)]$. Ultimately, Eq. **2** makes the transition from describing a material to designing a material in just one step.

Comparison Against Black-Box Optimizers

To demonstrate these strengths, we test our method against standard approaches that feed simulation parameters into a model by way of a black-box optimizer. We use adaptive simulated annealing (ASA) (29) and the CMA-ES (24). In each test, we allow the optimizers a fixed budget of material simulations, and each simulation requires a fixed amount of computational power. Thus, in our comparisons, computational cost and number of simulations are equivalent and an efficient optimizer is one that solves a design problem simulating as few candidate materials as possible. Implementation details are in *Supporting Information*.

As a first example, we task these two black-box algorithms and our approach with designing a square-lattice Ising model to maximize the magnitude of its magnetization. To do so, each method is allowed to vary the coupling constants that define the energy of spin alignments in the horizontal (J_x) and vertical (J_y) directions. To implement the black-box methods, we allow each optimizer to guess a set of coupling constants and evaluate the quality of that guess by computing the average magnetization. We find that both ASA and the CMA-ES struggle when searching entirely in the zero magnetization phase. This is an obvious consequence of the fact that the optimizer sees no variation in the quality for each parameter setting. Consequently, it receives no guidance about how to update its parameter guesses and can at best walk randomly until finding the phase boundary (Fig. 1 *A* and *B*).

By contrast, the updates encoded in Eq. **2** navigate a path that links one phase to the other: Fig. 1*C* shows the flow field generated by Eq. **2** upon taking $\rho(x|\lambda)$ to be the canonical ensemble with the Ising Hamiltonian, $H = -J_x \sum_{[ij]_x} s_i s_j - J_y \sum_{[ij]_y} s_i s_j$, where $[ij]_x$ denotes summing over nearest neighbors along the x direction, likewise for $[ij]_y$, and s_i denote the spin variables. Given this statistical model, the control parameters λ_i for optimization become $\lambda_x = J_x/kT$ and $\lambda_y = J_y/kT$. Finally, the quality function $f(s)$ is set to reward states with higher magnetizations (*Supporting Information*). If, for shorthand, we call the individual energy components $h_x = \sum_{[ij]_x} s_i s_j$ and $h_y = \sum_{[ij]_y} s_i s_j$, then Eq. **2** gives the velocity field $\dot{\lambda}_x = (1/|C|)(\mathrm{Cov}[h_y, h_y]\mathrm{Cov}[h_x, f] - \mathrm{Cov}[h_x, h_y]\mathrm{Cov}[h_y, f])$, where $|C| = \mathrm{Cov}[h_x, h_x]\mathrm{Cov}[h_y, h_y] - \mathrm{Cov}[h_x, h_y]^2$. A similar equation holds for $\dot{\lambda}_y$ but with the variables x and y appropriately interchanged.

In this form, it is clear that our method will optimize as long as there is covariation between the quality function, f, and the energy components, h_i. Because magnetization and energy are correlated, even if the average magnetization is zero, Eq. **2** can purposefully optimize even when operating in regions of parameter space where the black-box methods fail. The difference between these

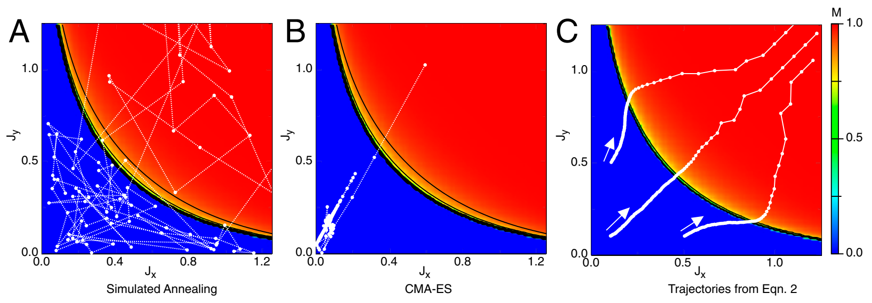

Fig. 1. Phase transitions and flat landscapes. If a materials design problem features a phase transition as part of the search space, black-box optimizers can struggle or fail due to inefficient use of simulation data. For example, if optimizing the magnetization, *M*, of a 2D Ising model by changing the coupling constants along the *x* and *y* directions (J_x/kT and J_y/kT, respectively), black-box methods like simulated annealing (*A*) and the CMA-ES (*B*) walk randomly if initialized in the zero magnetization state. Our method (*C*) interprets fluctuations in solution quality against model components as encoded by Eq. **2** and thus can navigate to the magnetized state regardless of initialization.

approaches lies in the fact that black-box methods are trying to solve a problem defined over the space of λ_i, whereas our approach is tasked to solve a problem defined on the space of configurations, x, via Eq. **1**. The CMA-ES generates multiple guesses of parameters from a Gaussian distributed over all possible λ and ASA samples by assigning an energy value to each choice of λ. These methods associate only one piece of information, the quality function, to full ensembles of configurations defined by each choice of parameters. This is in contrast to our approach that tries to solve the problem of reproducing configurations, x, that are better than average. Consequently, Eq. **2** is able to use information about how fluctuations in configurations correspond to fluctuations in quality because it has been built to exploit the extra fact that the simulation data were generated from a known distribution, ρ.

As a second example, we consider a thermalized particle trapped on a 2D substrate, defined on the $x_1 - x_2$ plane and at thermal equilibrium. The substrate applies a potential to the particle, making some positions more likely than others. We task the optimizer with trapping the particle in a specific potential well. To do so, we give the optimizer the freedom to tune the interaction strength with the substrate, and we give it control over a linear electric field to drive the particle. To make the problem interesting, we use a rough substrate potential: $h_s = \sum_i (-\cos[x_i(2\pi/5)] + x_i^2/25)$. With the field included, the total Hamiltonian becomes $H = h_s + v_{x_1}x_1 + v_{x_2}x_2$, where v_{x_1} and v_{x_2} represent the field strength in the two coordinate directions. To simplify the form of Eq. **2**, we represent these effects to the optimizer by defining $\lambda_s = 1/kT$ and absorb a factor of kT into the field coupling constants so that $\rho \propto \exp[-\lambda_s h_s - \lambda_{x_1}x_1 - \lambda_{x_2}x_2]$. In discussing the optimizer's performance, however, we will convert the results to the original, physical variables kT, v_{x_1}, and v_{x_2}.

The solution to this problem requires the design engine to make the target well the global minimum, and cool the system to zero temperature to trap the particle. For definiteness, the target well is the point (5,5) and we initialize the algorithm with the substrate at $1kT$ and the field parameters set to zero.

In Fig. 2*A* we plot the energy landscapes generated by the optimizer, as well as the points sampled during each iteration. Indeed, the optimizer quickly learns to tilt the landscape, correctly making the target well the global minimum, and then cools the system (Fig. 2*B*), trapping the particle in the well. By comparing the performance against black-box approaches (Fig. 2*C*), our approach is both faster and more reliable: It correctly tilts the well after only 35 iterations, whereas it takes ~ 100 simulations for the CMA-ES and ASA. Further, neither of the black-box methods learns to completely cool the system in the allotted 1,000 simulated ensembles.

We speculate that this shortcoming is again a consequence of indirect problem representation. We note that when $kT \approx 0.1$, the particle is almost evenly distributed between both the central well and the four nearest neighbors that surround it. Thus, for black-box methods, noise in the average particle position can play an overpowering role during parameter updates. By contrast, Eq. **2** considers covariances at the finer scale of configuration space. As with our prior example, this extra information makes our method appreciably more robust to flat regions in the search landscape and in this case yields an essentially exponential convergence to the optimized state (Fig. 2*B*).

Designing a Polymer to Fold into an Octahedron

The success of the two simple examples in the previous section invites more complicated design problems. As an example, we consider a basic model for a polymer: a string of hard, colored balls interconnected by rigid rods. The balls are weakly attractive, interaction strengths determined by the colors. For example, red and blue may be attracted more strongly than blue and green.

In principle, by tuning the color interactions, it should be possible to fold the chain into specific, desired shapes. To make a concrete task, we take a chain of six particles and create an optimizer to fold them into an octahedron, defined by minimizing the sum of the distances to the center of mass. Note that the search space is appreciably

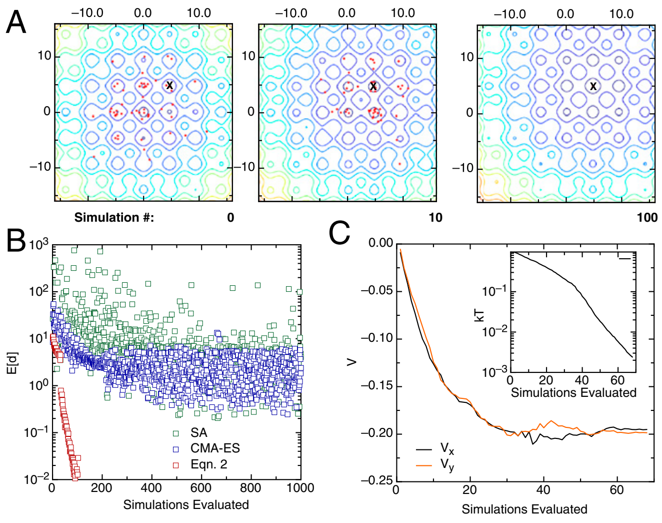

Fig. 2. Trapping a particle in a well. Here we treat the problem of a thermalized particle, trapped on a sinusoidal energy landscape superimposed on a quadratic background with a minima at the origin, depicted by the energy contours in *A*. The optimizer is given control over the system temperature kT plus a linear field potential in the two coordinates v_{x_1} and v_{x_2}. Its task is to trap the particle in a specific well located off center at (5,5) in the $x_1 - x_2$ plane, marked by a cross in *A*. The sampled particle locations, given each choice of parameters generated, after evaluating 0, 10, and 100 iterations of Eq. **2** are plotted as red points. The optimizer uses the field to first tilt the potential and make the target well the global minimum, and then it cools the system. In tasking the same problem to black-box optimizers, we find that both ASA and the CMA-ES are able to tilt the well, but never learn to cool the system (*B*). Comparing how the temperature and field parameters change at each iteration (*C*) against d_{ave} in *B* shows that the exponential convergence occurs in concert with an exponential decrease in system temperature. We note that the field parameters v_{x_1} and v_{x_2} track one another, reflecting the fact that the optimizer is invariant to rotations in the configuration space: The optimizer moves the field along the direction associated with the greatest improvement in solution quality and is insensitive to the fact that the problem was parameterized in the arbitrary coordinates x_1 and x_2.

larger than in the prior examples (dimension 10), and that simply setting all of the interaction strengths to large values will not produce the optimal solution. By choosing the interaction appropriately, the same energy can be given to the octahedral and polytetrahedral configurations for identical coupling constants. Entropic arguments imply that the polytetrahedron will dominate the chain configurations unless the optimizer carefully adjusts the coupling constants to take on unique values (30, 31).

Fig. 3*A* shows a typical chain configuration from each generation, and Fig. 3*B* shows the median sum of distances to the center of mass, normalized relative to that of a perfect octahedron. Initially, the coupling constants are set to $1kT$, and random chain configurations are typical. However, as the optimizer drives the interaction energies to larger values, the shapes become compact and structured. Around 200 generations, virtually every shape generated is octahedral (Fig. 3*A*).

By plotting the values of the interaction strengths against iteration number, we find the optimizer's solution is simple, logical, and arguably optimal. Early on, the optimizer attempts to meet the design goal by simply increasing the coupling strengths to make more compact objects (Fig. 3*A*). However, as the coupling constants are undifferentiated, the results are predominately polytetrahedral geometries. To compensate, the optimizer deactivates three coupling constants around 100 generations and sends the remainder to infinity (Fig. 3*C*). The logic behind this maneuver becomes clear by plotting the interactions as a network: The active interactions plus the polymer backbone form the contact graph of an octahedron. This strategy, transforming the contact matrix to an interaction matrix, has been identified as an approach to programing, by hand, the optimal interaction parameters for self-assembly (18). In fact, the specific problem of creating a self-assembling octahedron has been solved using a virtually identical motif (32). Evidently our optimizer can reproduce a well thought-out approach to self-assembly, and it does so automatically, requiring only a model and a design goal.

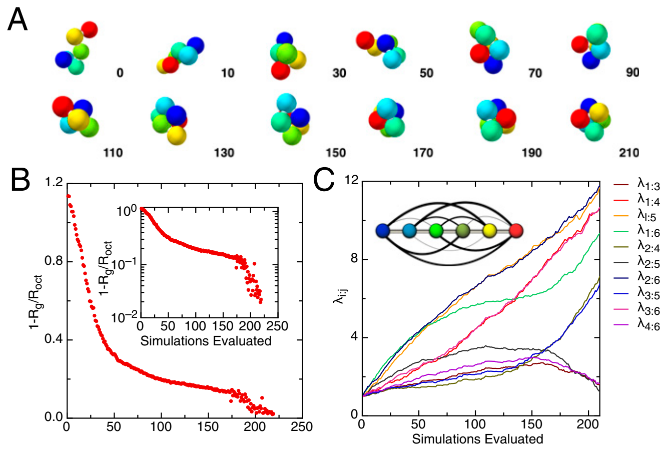

Fig. 3. Folding an octahedron out of a linear chain. (*A*) Typical chain configurations that result after iterating Eq. **2**. Numbers indicate the iteration number. Early on, the polymer configurations are dominated by thermal energy and random, yet as the interaction strengths increase and differentiate, more structured objects appear, and ultimately only the octahedron configuration exists (iterations 190–210). (*B*) Plotting the median percentage of deviation between the polymer's radius of gyration R_g and the radius of gyration for an octahedron R_{oct} at each iteration shows that the optimizer again produces a monotonic decrease at each step, with an effectively exponential convergence in the last 30 iterations. By the last 10 iterations, the median deviation from a perfect octahedron is roughly 1%. (*C*) In plotting the coupling constants against iteration number, we find that the optimizer adjusts coupling constants in groups. It is clear that the active coupling constants form the contact network for an octahedron (*C*, *Inset*).

Optimization of an Out-of-Equilibrium System

Because Eq. **2** holds for any parameterized probability distribution function, it can be used to create optimization schemes beyond the canonical ensemble in the prior examples. The only essential ingredients are a model that predicts the probability of microstates, an engine that samples configurations from said model, and a design goal. As simple extensions, chemical potentials or constraints on pressure could be included as tunable parameters (33). A new theoretical concept, termed "digital alchemy," extends statistical mechanics to account for microscale geometric parameters, such as the particle shape in a colloidal-nanoparticle assembly (34). Thus, by coupling this approach with our optimization formalism, particle geometry can be tuned to produce optimized bulk responses. One can also note that the range of parameters to design is at the user's discretion: Eq. **1** can be used to rederive Eq. **2**, assuming that some of the model parameters are not controllable by taking them to be time independent. Indeed, for the particle in a well problem, the wavelength of corrugation was taken as a fixed parameter and the resulting optimizer was quite effective. One can also consider optimization for global quality functions that exist over multiple a range of parameters (35). For instance, our approach can optimize the density of a crystal lattice over a range of pressures and system volumes by defining $\rho=\rho_0(x|\lambda,V,P)U(V,P)$, where $U(V,P)$ is a uniform distribution for V and P over a range of consideration and ρ_0 is the appropriate distribution for microstates given a fixed volume and pressure. Eq. **2** can then be applied to optimize in this extended parameter space to find choices of λ that work well over a range of possible densities and pressures. Abstracting the concept, statistical models could be based on calculations like self-consistent field theory or experiments with the code directly measuring correlation functions in the laboratory and tuning physical parameters.

Moreover, nonequilibrium processes, provided they have a statistical description, are fair game. For example, if one simulates diffusion by adding white noise to a mean drift, then the paths are distributed by a product of Gaussian distributions conditioned on the prior steps. Clearly, the paths are statistical objects, with diffusion and drift as the distribution parameters, λ. Thus, one can build an optimizer that tunes these control parameters using Eq. **2**, even if they are time-dependent functions.

As proof of this point, we return to the problem of a particle trapped on a substrate, but now simulate the particle dynamics. The applied field and the system temperature are treated as functions of time and the optimizer is tasked with moving the particle from one well to another in a given interval.

Fig. 4*A* shows the median distance to the target well after executing the optimizer's processing protocol in each generation. In the first 60 generations, the optimizer learns to transport the particle from its starting location to the target well via a large, deterministic driving force. It then spends the remaining iterations monotonically decreasing the system temperature while developing a trapping protocol with the field. After 2,000 iterations, the optimizer seems to traps the particle by oscillating the driving force, changing its direction before the particle can transition to another well. In effect, the optimizer learns to drag the particle to the target and trap it in place, using both the temperature and the field. By 2,000 iterations, $\sim 90\%$ of the points in the path fall within the target well. When left to run longer, the optimizer continues to improve the quality of solutions, but at the cost of becoming unphysical, the optimizer generates extremely large fields that move the particle to the well faster and faster. To optimize beyond the proof of concept demonstrated here, one may have to restrict the range of parameters allowed to the optimizer or account for arbitrary velocities by increasing the number of steps in the walk.

Optimization of Directed Self-Assembly

As a final demonstration of our approach we consider the real-world problem of designing the directed self-assembly of block copolymers on a chemically patterned substrate. This represents a task at the forefront of both materials design and sublithographic

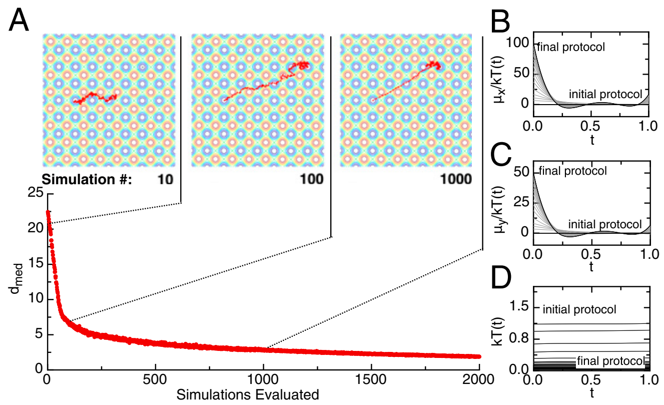

Fig. 4. Optimizing a nonequilibrium process. By using Eq. **2** we can tune processing protocols for a Brownian particle walking on a rough energy landscape controlled by a time-dependent temperature, kT, and linear mean drift components, μ_x and μ_y. The optimizer has been tasked to adjust to place and trap the random walker in a well located at the x–y coordinates (10,10). (*A*) Ensemble median distance to the objective well after executing a processing protocol at each iteration of the algorithm. Callouts show representative paths taken by the particle, and contours in the callouts show lines of constant energy over the substrate potential. The large image represents the protocol executed after 2,000 iterations. Every protocol attempted at 10-iteration intervals is illustrated in *B*–*D*, where the temperature, kT (*D*) as well as the applied fields in the x direction and y direction normalized by temperature, μ_x/kT (*B*) and μ_y/kT (*C*), are plotted against time. At $t=0$ the particle is released from its initial position at $(-10,0)$ and allowed to wander and the processing protocol is executed until the simulation is stopped at $t=1$. At each iteration, the optimizer works to monotonically decrease the temperature, while arriving at a field protocol that quickly drives the particle to the target well and then oscillates the fields to trap it there.

fabrication (1, 7–11, 36, 37). The goal is to lithographically pattern a substrate with a small number of chemical features such that these features promote block copolymers to self-assemble into a desired target morphology. Here, we consider a task that has been identified as a promising candidate for the manufacture of next-generation semiconductor devices and high-density storage media: self-assembly of AB-diblock copolymers into an ordered striped or lamellar morphology (1, 9–11).

We use a theoretically informed coarse-grain model for block copolymer simulations (36). Polymer chains are simulated as beads that are linked together. The system is considered at fixed temperature and volume, and thus the probability of finding a given microstate configuration is defined in terms of an energy given by three parts. The first is a linear spring bond energy between beads in each polymer chain. The second is a nonbonded interaction energy that characterizes repulsion from unlike species and the material compressibility. Details are in *Supporting Information* (36). Both the model and the parameters in it represent a polystyrene-block-poly(methyl methacrylate) (PS-b-PMMA) diblock copolymer with a number-averaged molecular weight of 22,000-b-22,000 and a stripe period of 28 nm.

The final contribution to the energy is the substrate interaction. The substrate consists of two regions: the patterned stripes of width w and the background. Both are defined to have short-ranged effects on the polymer beads and assume the form $H_s/kT=\Lambda(\alpha)/d_s\exp[-(z/2d_s)^2]$, where d_s defines the decay length of the interaction, z is the distance from the plane of the substrate, and $\Lambda(\alpha)$ is the interaction strength between the substrate and a bead of type α. Thus, if the particle is over the guiding stripe and of type A, $\Lambda(A)=\Lambda_s$. If the particle is of type A and over the background region, $\Lambda(A)=\Lambda_b$. Following ref. 11, we simplify our model by assuming the interactions are antisymmetric: $\Lambda_s(A)=-\Lambda_s(B)$ and $\Lambda_b(A)=-\Lambda_b(B)$. The design problem is to adjust the width of the strips w and the two energy parameters Λ_s and Λ_b so the target stripe phase replicates itself m times between two guiding stripes spaced by the polymer period multiplied by m.

The results for $m=3$ and $m=6$ density multiplication are shown in Fig. 5. For the $m=3$ problem, we ran the optimization four times, varying the time step used in integrating Eq. **2**. In every instance, the optimizer not only brought the system to a state that successfully met the design goal, but also, within noise, converged to the same state. The optimized parameters suggest that directed self-assembly is best achieved by setting the stripe width equal to roughly half the polymer period, $\Lambda_s\approx-1kT$ and $\Lambda_b=0.05kT$. All of these parameters agree with simulation results obtained by a brute-force solution to the problem and experimental verifications performed on the real polymer system (37) and are physically consistent with optimization results obtained for triblock copolymer pattern multiplication (11). These results can be explained by considering the interfacial energies in the system. The background interaction is required to be weak because the background region has roughly equal coverage between the A and B phases and is significantly larger in area than the size of the stripe. Moreover, the interaction strength for the stripe components is larger in the $m=6$ problem (Fig. 5*D*), because the larger distance between patterned regions requires stronger anchoring to guide assembly.

When the optimizer was run at the most aggressive time step, we were able to achieve convergence for m = 3, (m = 6) in less than 10 (18) iterations, despite the fact that the material required the simulation of roughly 50,000 (100,000) polymer beads. We stress that the performance obtained here is not a consequence of initializing the system too close to an optimal state, but rather evidence of the power behind Eq. **2**. Fig. 5 shows that indeed the

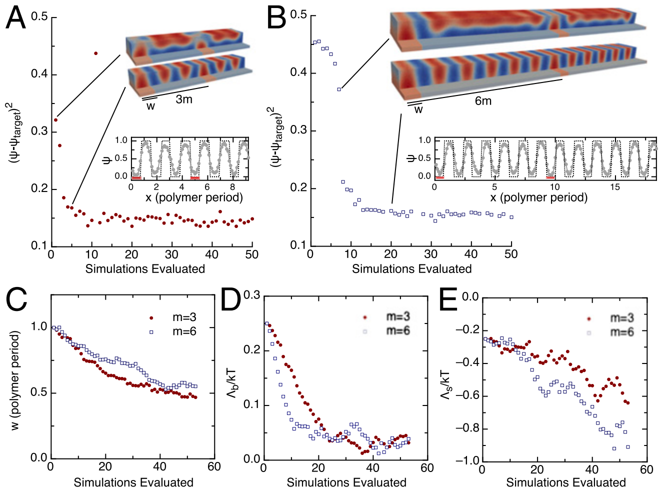

Fig. 5. Optimizing directed self-assembly. Here we design the width of the stripe, the strength of its attraction to the red polymer beads Λ_s, and the attraction strength of the background substrate Λ_b toward the blue polymer beads, to match the self-assembled phase as closely as possible to the target of alternating stripes. We quantify the success of our optimizer by comparing an order parameter $\Psi(x)=(n_a)/(n_a+n_b)$ binned along the x axis of the box and averaged over y and z to the target stripe pattern. With Eq. **2**, we are able to produce optimized parameters for 3× (*A*) and 6× (*B*) density multiplication after simulating 10 and 20 parameter choices. Two characteristic configurations are plotted in *A* and *B*, *Insets*, separated by just a handful of iterations, yet displaying markedly different phases of the polymer. Asymptotic configurations depicting Ψ (solid, marked line) and the target (dashed line) (*A* and *B*, *Insets*) show that the optimized parameters match the desired morphology, typically within 80% or better. In plotting the parameters generated by Eq. **2** (*C*–*E*), we find that the interaction with the background brush is the most relevant parameter in directed self-assembly. For both the 6× and 3× problems, the rapid convergence toward the optimized state takes place once the background strength is reduced to $\Lambda_b\approx0.05$. After a weak background is established, the strip width and strength function as fine-tuning parameters that facilitate defect-free assembly.

initial parameters do not produce a solution to the design problem, let alone a stripe pattern of any kind.

This particular problem also has been attempted in some variant using other materials design methods. In fact, our initial conditions were selected to match those given to an implementation of the CMA-ES, solving the same design problem but using triblock copolymers (11). For that problem, the CMA-ES took roughly 50 generations to converge, simulating 32 ensembles in parallel per iteration, each requiring 200,000 Monte Carlo steps. Our approach also used 200,000 Monte Carlo steps per iteration, but required only 10 iterations to converge. If algorithm performance is measured in terms of the number of microstates simulated, then solving directed self-assembly problems by way of the CMA-ES requires at least 5 times as much compute power as the approach proposed here. If it is not possible to run the ensemble simulations in parallel, our approach is roughly 130 times faster than the CMA-ES and completes a full optimization process before the CMA-ES has completed a single iteration. Additionally, inverse Monte Carlo methods have been used to solve directed self-assembly problems involving the placement of guiding posts instead of stripes (8). Although there are relevant physical differences, we note that the results presented for inverse Monte Carlo converge after simulating roughly 30 million microstates. Because this number of microstates simulated is roughly 15 times larger than what was used here, we can speculate that our proposed methods could also be faster for such applications.

Conclusions

To the extent that the goal of materials design is a unified framework that handles a wide range of complex inverse problems, we believe the formalism introduced here represents a significant step forward. By applying Eq. **2**, we can solve problems with flat search landscapes (Fig. 1) and multiple-interaction types (Fig. 2), incorporate constraints (Fig. 3), tune processing conditions (Fig. 4), and address application scale design and optimization tasks (Fig. 5). Furthermore, in all of the examples presented, the end result is intuitive even though it was achieved in a complicated search landscape where other optimization schemes struggle or fail. Finally, the fact that processing conditions such as applied fields or temperature protocols and model parameters like internal interaction energies can be optimized with the very same framework presents an unexplored direction for materials design. Because these are the essential aspects that determine the properties of any material, the capacity to tune both simultaneously, one accounting for the other, opens the doors to a more coherent and conceptually complete design program.

ACKNOWLEDGMENTS. We thank Arvind Murugan, Jim Sethna, Sid Nagel, Suriyanarayanan Vaikuntanathan, and Tom Witten for many insightful discussions, and the reviewers for highly constructive suggestions. This work was supported by the National Science Foundation through Grant CBET 1334426. We acknowledge additional support through award 70NANB14H012 from the US Department of Commerce, National Institute of Standards and Technology as part of the Center for Hierarchical Materials Design (CHiMaD).

1. Jain A, Bollinger JA, Truskett TM (2014) Inverse methods for material design. *Am Inst Chem Eng J* 60(8):2732–2740.
2. Torquato S (2009) Inverse optimization techniques for targeted self-assembly. *Soft Matter* 5(6):1157–1173.
3. Jaeger HM (2015) Celebrating Soft Matter's 10th anniversary: Toward jamming by design. *Soft Matter* 11(1):12–27.
4. Oganov AR, Glass CW (2006) Crystal structure prediction using ab initio evolutionary techniques: Principles and applications. *J Chem Phys* 124(24):244704.
5. Dahiyat BI, Mayo SL (1997) De novo protein design: Fully automated sequence selection. *Science* 278(5335):82–87.
6. Kuhlman B, et al. (2003) Design of a novel globular protein fold with atomic-level accuracy. *Science* 302(5649):1364–1368.
7. Hannon AF, et al. (2013) Inverse design of topographical templates for directed self-assembly of block copolymers. *ACS Macro Lett* 2(3):251–255.
8. Hannon AF, Ding Y, Bai W, Ross CA, Alexander-Katz A (2014) Optimizing topographical templates for directed self-assembly of block copolymers via inverse design simulations. *Nano Lett* 14(1):318–325.
9. Chang J-B, et al. (2014) Design rules for self-assembled block copolymer patterns using tiled templates. *Nat Commun* 5:3305.
10. Qin J, et al. (2013) Optimizing directed self-assembled morphology. *Soft Matter* 9(48): 11467–11472.
11. Khaira GS, et al. (2014) Evolutionary optimization of directed self-assembly of triblock copolymers on chemically patterned substrates. *ACS Macro Lett* 3(8): 747–752.
12. Bianchi E, Doppelbauer G, Filion L, Dijkstra M, Kahl G (2012) Predicting patchy particle crystals: Variable box shape simulations and evolutionary algorithms. *J Chem Phys* 136(21):214102.
13. Miskin MZ, Jaeger HM (2013) Adapting granular materials through artificial evolution. *Nat Mater* 12(4):326–331.
14. Miskin MZ, Jaeger HM (2014) Evolving design rules for the inverse granular packing problem. *Soft Matter* 10(21):3708–3715.
15. DiStasio RA, et al. (2013) Designer spin systems via inverse statistical mechanics. *Phys Rev B* 88:134104.
16. Solis EOP, Barton PI, Stephanopoulos G (2010) Formation of nanostructures with desired geometries. 1. Robust static structures. *Ind Eng Chem Res* 49(17):7728–7745.
17. Jankowski E, Glotzer SC (2012) Screening and designing patchy particles for optimized self-assembly propensity through assembly pathway engineering. *Soft Matter* 8:2852–2859.
18. Hormoz S, Brenner MP (2011) Design principles for self-assembly with short-range interactions. *Proc Natl Acad Sci USA* 108(13):5193–5198.
19. Landau LD, Lifshitz EM (1980) *Statistical Physics. Part 1: Course of Theoretical Physics* (Pergamon), 3rd Ed, pp 333–368.
20. Amari S-I (1998) Natural gradient works efficiently in learning. *Neural Comput* 10(2): 251–276.
21. Wierstra D, Schaul T, Peters J, Schmidhuber J (2008) Natural evolution strategies. *Evolutionary Computation (IEEE World Congress on Computational Intelligence)* (IEEE), pp 3381–3387.
22. Ollivier Y, Arnold L, Auger A, Hansen N (2011) Information-geometric optimization algorithms: A unifying picture via invariance principles. *arXiv preprint*:1106.3708.
23. Akimoto Y, Nagata Y, Ono I, Kobayashi S (2010) Bidirectional relation between CMA evolution strategies and natural evolution strategies. *Parallel Problem Solving from Nature, PPSN XI*, eds Schaefer R, Cotta C, Kołodziej J, Rudolph G (Springer, Berlin), pp 154–163.
24. Hansen N, Müller SD, Koumoutsakos P (2003) Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evol Comput* 11(1):1–18.
25. Transtrum MK, Machta BB, Sethna JP (2011) Geometry of nonlinear least squares with applications to sloppy models and optimization. *Phys Rev E Stat Nonlin Soft Matter Phys* 83(3 Pt 2):036701.
26. Cressman R (2005) Stability of the replicator equation with continuous strategy space. *Math Soc Sci* 50(2):127–147.
27. Oechssler J, Riedel F (2002) On the dynamic foundation of evolutionary stability in continuous models. *J Econ Theory* 107(2):223–252.
28. Weibull JW (1997) *Evolutionary Game Theory* (MIT Press), pp 69–121.
29. Ingber L, et al. (2012) Adaptive simulated annealing. *Stochastic Global Optimization and Its Applications with Fuzzy Adaptive Simulated Annealing*, eds Junior HAeO, Ingber L, Petraglia A, Petraglia MR, Machado MAS (Springer, Berlin), pp 33–62.
30. Arkus N, Manoharan VN, Brenner MP (2009) Minimal energy clusters of hard spheres with short range attractions. *Phys Rev Lett* 103(11):118303.
31. Meng G, Arkus N, Brenner MP, Manoharan VN (2010) The free-energy landscape of clusters of attractive hard spheres. *Science* 327(5965):560–563.
32. Zeravcic Z, Brenner MP (2014) Self-replicating colloidal clusters. *Proc Natl Acad Sci USA* 111(5):1748–1753.
33. Ferrenberg AM, Swendsen RH (1988) New Monte Carlo technique for studying phase transitions. *Phys Rev Lett* 61(23):2635–2638.
34. van Anders G, Klotsa D, Karas AS, Dodd PM, Glotzer SC (2015) Digital alchemy for materials design and optimization. *arXiv preprint*:1507.04960.
35. Jain A, et al. (2013) Inverse design of simple pairwise interactions with low-coordinated 3d lattice ground states. *Soft Matter* 9(14):3866–3870.
36. Detcheverry FA, et al. (2008) Monte Carlo simulations of a coarse grain model for block copolymers and nanocomposites. *Macromolecules* 41(13):4989–5001.
37. Liu CC, et al. (2013) Chemical patterns for directed self-assembly of lamellae-forming block copolymers with density multiplication of features. *Macromolecules* 46(4): 1415–1424.
38. Wolff U (1989) Collective Monte Carlo updating for spin systems. *Phys Rev Lett* 62(4): 361–364.
39. Baumgärtner A (1980) Statics and dynamics of the freely jointed polymer chain with Lennard-Jones interaction. *J Chem Phys* 72(2):871–879.
40. Baumgärtner A, Binder K (1979) Monte Carlo studies on the freely jointed polymer chain with excluded volume interaction. *J Chem Phys* 71(6):2541–2545.
41. Earl DJ, Deem MW (2005) Parallel tempering: Theory, applications, and new perspectives. *Phys Chem Chem Phys* 7(23):3910–3916.
42. Abramowitz M, Stegun IA (1972) *Handbook of Mathematical Functions: With Formulas, Graphs, and Mathematical Tables* (Dover, New York), 9th Ed.