Continual Learning of Multiple Memories in Mechanical Networks

Menachem Stern,[1] Matthew B. Pinson,[1,2] and Arvind Murugan[1,*]

[1]*Physics Department and the James Franck Institute, University of Chicago, Chicago, Illinois 60637, USA*
[2]*Newman College, University of Melbourne, Parkville, Victoria 3052, Australia*

(Received 28 May 2020; accepted 29 June 2020; published 25 August 2020)

Most materials are changed by their history and show memory of things past. However, it is not clear when a system can continually learn new memories in sequence, without interfering with or entirely overwriting earlier memories. Here, we study the learning of multiple stable states in sequence by an elastic material that undergoes plastic changes as it is held in different configurations. We show that an elastic network with linear or nearly linear springs cannot learn continually without overwriting earlier states for a broad class of plasticity rules. On the other hand, networks of sufficiently nonlinear springs can learn continually, without erasing older states, using even simple plasticity rules. We trace this ability to cusped energy contours caused by strong nonlinearities and thus show that elastic nonlinearities play the role of Bayesian priors used in sparse statistical regression. Our model shows how specific material properties allow continual learning of new functions through deployment of the material itself.

DOI: 10.1103/PhysRevX.10.031044

Subject Areas: Metamaterials, Soft Matter, Statistical Physics

I. INTRODUCTION

Multistability is an emergent nonlinear behavior found in systems as diverse as electrical, neural, biochemical, and mechanical networks [1]. In particular, multistability has been sought in mechanical systems as a way to achieve multifunctionality and has been engineered in sheets, shells, and, more generally, elastic networks [2–13]. But in all of these examples, the mechanical metamaterial or elastic network is carefully constructed with *a priori* knowledge of all the desired states. Hence, adding a new stable state typically requires rewiring the entire network from scratch.

In contrast, many networks in the natural world are grown over time according to local rules and are thus naturally shaped by the current geometry of the system [14–16]. For example, actomyosin networks grow between focal adhesion points of a cell; different focal adhesion geometries naturally result in different networks with different properties [17]. Examples of networks grown according to local geometry are found on all scales, from microtubules [18] and synthetic DNA nanotubes grown between molecular landmarks [19,20] to tissues [21] and even entire macroscopic organisms like *Physarum* [22]. Such organic growth of networks raises the possibility of a mechanical network acquiring new stable states—and thus new functionality—on the fly through incremental changes, without rewiring from scratch to include each new state.

While such continual learning is appealing, a primary challenge is that each learned behavior needs to survive changes during learning of subsequent behaviors and not be overwritten. The requirements for such continual learning without erasure have been studied in neural networks since Hopfield [23] and Gardner [24], and from an active area of research [25], but the requirements for continual learning of mechanical behaviors are not clear.

In this work we study the requirements for continual learning of multiple stable states in a simple elastic network. We first study a concrete model of continual learning, motivated by networks grown over time in nature. In this model, the material is placed in each of the desired states in sequence for a period of time. During this time, elastic rods or springs with a rest length grow between particles within some distance in space, mimicking the seeded growth of microtubules [18] or self-assembling DNA nanotubes [19]. Thus, this learning model is constrained by locality in space and time—material parameters are modified only by the local geometry of the current configuration being experienced [26,27,48].

We find that continual learning of new states without forgetting the old requires nonlinear elasticity of a specific type. Parametrizing the elastic energy of springs in the network as $E \sim s^{\xi}$ for large strain s, continual learning requires $0 < \xi \leq 1$. Nonlinear springs $0 < \xi \leq 1$ have been

*Corresponding author.
amurugan@uchicago.edu

demonstrated using metamaterials such as origami [28,29]. In contrast, if all desired states are known *a priori* and continual learning is not desired, linear springs $\xi = 2$ (Hooke's law) are sufficient to stabilize multiple states.

We then generalize beyond our simple model of plasticity; we show that such $\xi < 1$ mechanical nonlinearities are in fact required of any model of continual learning in which parameters are incrementally updated with knowledge only of the current desired state and no knowledge of future or past states. Our general results relate the distinction between $\xi > 1$ springs and $\xi < 1$ nonlinear springs to smooth and cusped energy contours, respectively. We show that, consequently, networks of linear springs share strain democratically, but networks of $\xi \leq 1$ nonlinear springs show winner-take-all behavior; some springs can be nearly unstrained while others are highly strained. In this way, we argue that $\xi \leq 1$ mechanical nonlinearities play the role of nonlinear Bayesian priors used in sparse regression.

Finally, we discuss natural and synthetic materials that could form the basis for continually learning networks. We hope our analysis of a simple mechanical model will stimulate further work on the conditions under which materials can learn new functionalities on the fly.

II. RESULTS

We seek to create an elastic network of springs connecting N particles in two dimensions, such that the network has M desired stable states [Fig. 1(a)]. Each desired state $m = 1, \ldots, M$ is specified by the positions $\mathbf{x}^{(m)}$ of the N particles (up to rigid body translations and rotations).

In standard approaches, all desired stable states are known in advance and used for constructing the network [13]. As an example of such a framework, we connect the N particles by Hookean (linear) springs and solve an optimization problem for spring constants k_{ij} and rest lengths l_{ij} that minimizes residual forces at each of the desired configurations $\mathbf{x}^{(m)}$ [Fig. 1(b)]; see the Appendix A for details. Note that in this model, adding a single new stable state requires a complete rewiring of the network with new Hookean springs.

For continual learning, we first consider a particular simple model of grown networks in which desired stable states are acquired by sequentially placing the material in the desired configurations [Fig. 1(c)]. We find that continual learning without forgetting requires nonlinear springs. Later, we show that such nonlinearities are in fact a requirement for a broad class of continual models in which the spring network is updated based on the current configuration alone, without knowledge of future or past desired configurations.

A. Simplified growth model

In our incremental growth model for continual learning, when the material is left in a configuration $\mathbf{x}^{(1)}$ for a length of time, unstretched elastic rods grow between every pair of particles i, j at a rate $f(r_{ij})$ set by their separation r_{ij}; we assume that f vanishes rapidly outside of a characteristic length scale R, so only nodes within a distance less than R are stabilized by such rods. Since the number of rods grows with time, the effective spring constant for the set of rods connecting two particles i, j grows with time and is given by

$$\frac{dk_{ij}^{\text{eff}}}{dt} = k_0 f(r_{ij}). \tag{1}$$

Here, k_0 is the spring constant of each rod, whose rest length l_{ij} is assumed equal to the particle separation r_{ij}; i.e.,

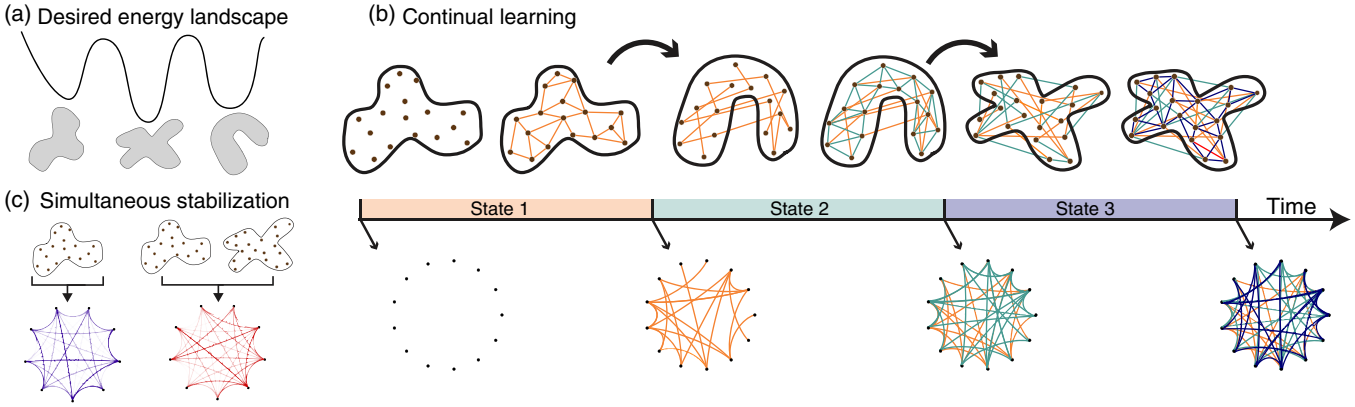

FIG. 1. Continually learned multistability. (a) We seek to create an elastic network with specific stable configurations (energy minima). (b) In continual learning, the network grows incrementally; new bonds created in one epoch (e.g., purple) can only depend on the configuration (e.g., state 3) during that epoch and have no information about states in the past and the future. For continual learning of new states without forgetting older states, network changes due to learning state 3 should not interfere with the stability of state 1 and 2 or vice versa. (c) In contrast, in the usual approach, all desired states are specified beforehand and then network parameters (e.g., connectivity, spring constants) are optimized to simultaneously stabilize these states. Unlike with continual learning, adding more states requires redesigning the network from scratch.

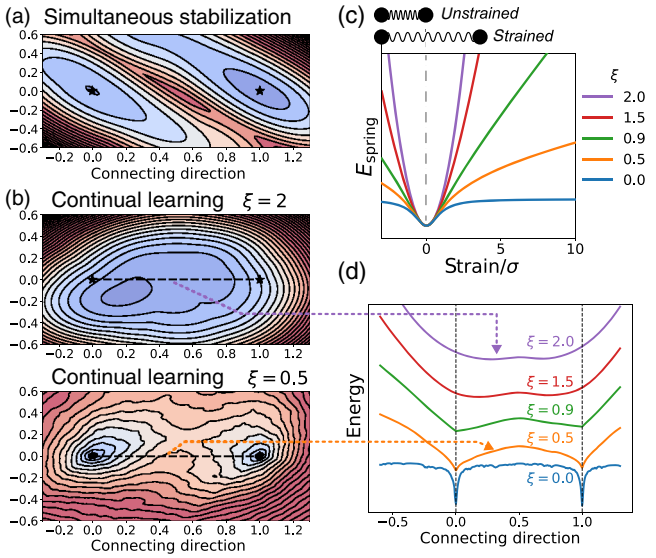

FIG. 2. Nonlinear interactions are essential for learning multiple states in sequence. (a) Energy landscape of a simultaneously stabilized network with linear ($\xi = 2$) springs, successfully stabilizing 2 desired states (black stars). (b) In the continually learned network, linear ($\xi = 2$) springs learned for each desired state destabilize the other state, but nonlinear ($\xi = 0.5$, $\sigma = 5 \times 10^{-3} L$) learned springs stabilize both desired states. (c),(d) Repeating learning for nonlinear springs with $E \sim s^{\xi}$, we find that learned states overwrite each other for $\xi > 1$ but are protected from each other with sufficiently nonlinear $\xi \leq 1$ springs.

rods are unstretched in the desired state. In simulations, we pick f to be a step function of range R, $f(r < R) = 1$, $f(r > R) = 0$. Our results below hold qualitatively for both short-ranged and long-ranged choices of R. Networks grown in this manner are seen in living systems (e.g., microtubules growing between centrosomes and centromeres [18,30]) and in synthetic systems (e.g., self-assembling DNA nanotubes [19] growing between seeds). To continually encode multiple stable states, we reshape the network into successive geometric configurations, letting new rods grow according to Eq. (1) in each one of the desired states. Note that since all grown rods are unstrained in the concurrent geometric state, only information about that current state is required to continually grow the network.

We ran both the above continual learning (growth) model and standard simultaneous stabilization described earlier on a pair of randomly generated desired state $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ of 10 particles. With linear Hookean springs ($E \sim ks^2$), the simultaneous stabilization algorithm successfully stabilizes both states; see Fig. 2. However, continual learning of the same two states $\mathbf{x}^{(1)}$, $\mathbf{x}^{(2)}$ fails. Rods grown to encode state $\mathbf{x}^{(1)}$ destabilize, or overwrite, state $\mathbf{x}^{(2)}$ and vice versa.

We then considered springs with nonlinear energies,

$$E(s) \sim k_0 \frac{s^2}{(\sigma^2 + s^2)^{1-(1/2)\xi}}, \tag{2}$$

where k_0 is the spring constant and $s_{ij} \equiv (r_{ij} - l_{ij})$ is the strain relative to rest length l_{ij}. ξ parametrizes the nonlinearity [Fig. 2(c)]; $\xi = 2$ is a linear Hookean spring while $\xi < 2$ springs have softer restoring forces at large distances, $E \sim s^{\xi}$. Finally, σ is a small length scale within which the interaction is linear for any ξ and is introduced to reflect practical realizations of nonlinear $\xi < 2$ springs [28,29]; our results continue to hold as $\sigma \to 0$. See Appendix B for details.

We repeated the same continual learning procedure on the same states as earlier—but with nonlinear springs $\xi < 2$. While the results for $1 < \xi < 2$ are qualitatively similar to Hookean $\xi = 2$ springs, $\xi < 1$ shows qualitatively different results—multiple states are stabilized in sequence [Figs. 2(b) and 2(d)].

The quality of learned states can be quantified by the attractor size and barrier heights around stored states. Large attractors and high energy barriers allow robust retrieval of states from a larger range of initial conditions. These measures have long been used to quantify quality of learning in neural networks [31–33]. We find that quality of learned and simultaneously stabilized states, as measured by barrier heights, is highest at distinct ξ^*; see Figs. 3(a) and 3(b). The quality of the simultaneously stabilized states is optimal for linear springs $\xi^* \approx 2$, relatively insensitive to the number of desired states. However, the optimal ξ^* for learned states is $0 < \xi^* < 1$ and varies with the number of learned states. We find similar results by measuring attractor radius instead of

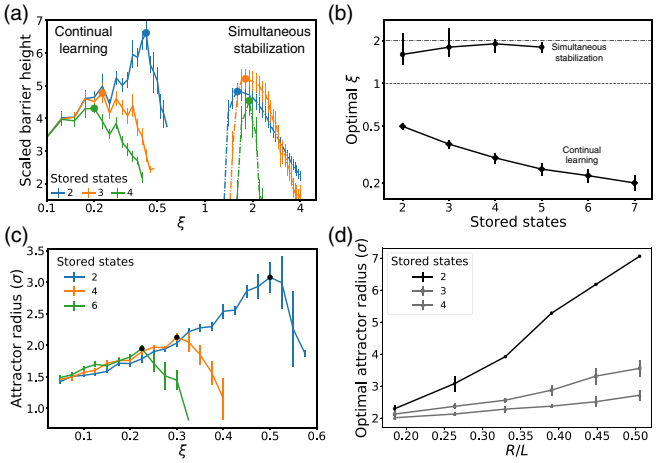

FIG. 3. Optimal nonlinearity for continually learned states. (a),(b) Barrier heights around learned states are highest at a specific nonlinearity $0 < \xi^* < 1$. Further, learning more states requires stronger nonlinearity ξ^* (compared to simultaneously stabilized sates that are most stable with for $\xi^* \approx 2$, or linear springs). (c) We find similar results by quantifying learning quality by attractor size around stable states. (d) Learning rules that connect more distant nodes, i.e., larger range R for $f(r)$ in Eq. (1), lead to larger attractor basins (see Appendix D for details). L is the system length, $\sigma = 5 \times 10^{-3} L$. Results averaged over 45 simulations of $N = 100$ particle networks.

barrier heights [Fig. 3(c)]. These results average over simulations of 45 systems with $N = 100$ particles at every value of ξ. As seen in Fig. 3(d), continual learning works for a range of spring connection distance R relative to system size L, though attractor size is larger for larger R/L. See Appendixes C and D.

III. CONTINUAL LEARNING REQUIRES NONLINEARITIES

Do our results hold beyond the simple growth model in Eq. (1)? Here, we define continual learning more broadly to include frameworks in which the spring network (e.g., spring constants, spring lengths) can change in any complex manner based on the current configuration, but the changes cannot depend on any knowledge of future or past configurations. We argue that this broader class of models also requires sufficiently nonlinear springs [$\xi \leq 1$ in Eq. (2)] for continual learning of multiple stable states.

For any kind of spring network, changes in the spring parameters due to being held in, say, state $\mathbf{x}^{(2)}$ will generally create unbalanced net residual forces $\mathbf{F}_2^{\text{erase}}(\mathbf{x})$ at any other configuration $\mathbf{x}$. In particular, these forces will not generically vanish at the previously stable configuration $\mathbf{x}^{(1)}$ since, by definition of continual learning, the spring network modifications while being held in state $\mathbf{x}^{(2)}$ cannot be cognizant of past or future states. Hence the net forces $\mathbf{F}_2^{\text{erase}}(\mathbf{x}^{(1)})$ at state $\mathbf{x}^{(1)}$ due to network changes to encode state 2 will always be nonzero and tend to destabilize and thus erase state 1.

However, we argue that these forces $\mathbf{F}_2^{\text{erase}}(\mathbf{x}^{(1)})$ generically distort state 1, $\mathbf{x}^{(1)} \to \mathbf{x}^{(1)} + \delta\mathbf{x}$, by a large amount $\delta\mathbf{x}$ for $\xi > 1$ springs, but $\delta\mathbf{x}$ is only infinitesimal for $\xi \leq 1$ springs because stable configurations are found at cusps of energy contours for $\xi \leq 1$.

To see this, consider the case of an unstrained spring in 1D, as in Fig. 4(a) (with $\sigma \to 0$, a small core length of linear response near the unstrained state). A spring with $\xi > 1$, on which an external force F^{erase} is applied, will extend by $\delta x \sim (F^{\text{erase}})^{1/(\xi-1)}$. In contrast, a $\xi \leq 1$ spring would extend by $\delta x \ll \sigma$ [Fig. 4(b)], provided $F^{\text{erase}} < F_T$. We find the threshold force to be $F_T \sim \sigma^{\xi-1}$ [Fig. 4(c)]. Consequently, the perturbation of spring length due to a perturbative force F is dramatically different for $\xi > 1$ and $\xi \leq 1$. See Appendix E for details.

Now we generalize this argument to spring networks. The state $\mathbf{x}^{(1)}$ is stabilized by a set of springs K_1 whose net force vanishes at that state. The net energy of this set K_1 for perturbations around $\mathbf{x}^{(1)}$ can be written as $\sum s_i^\xi$, where s_i is the strain in spring i, if all springs in K_1 were unstrained in state $\mathbf{x}^{(1)}$. For $\xi > 1$, with such an energy function, a force of magnitude F will perturb $x^{(1)}$ by $\delta x \sim F^{1/\xi-1}$ that grows with F. If the springs in K_1 were strained to begin with in state $\mathbf{x}^{(1)}$, the state is even more unstable.

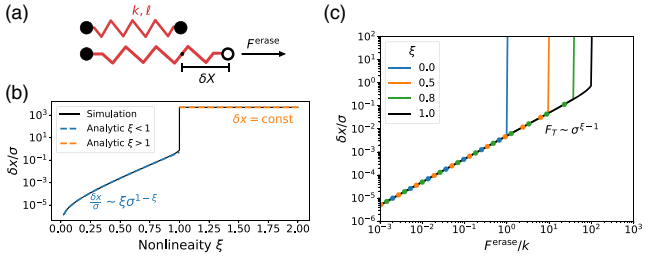

FIG. 4. Nonlinear $\xi \leq 1$ springs are essential for continual learning. (a) An unstrained spring is extended by δx due to an external force F^{erase}. (b) Net displacement δx of a previously stable point $\mathbf{x}^{(1)}$ due to forces F^{erase} arising from springs learned for a new state $\mathbf{x}^{(2)}$. $\delta x \ll \sigma$ for $\xi \leq 1$ springs, yet δx is extended for $\xi > 1$ springs. (c) A threshold erasure force $F_T \sim \sigma^{\xi-1}$ is required to destabilize the stable state due to nonlinear $\xi \leq 1$ springs. Small σ and $\xi \leq 1$ springs are thus essential for stabilizing previous states under continual learning of new states ($\sigma = 10^{-4}L$).

On the other hand, with $\xi \leq 1$ springs, the energy $\sim \sum s_i^\xi$ in the vicinity of $\mathbf{x}^{(1)}$ is cusped (in the limit $\sigma \to 0$); consequently, the restoring force due to springs in K_1 can be arbitrarily large for small displacements away from $\mathbf{x}^{(1)}$. A finite σ limits these restoring forces to a maximum value $F_T \sim \sigma^{\xi-1}$. Consequently, for erasure forces F^{erase} less than F_T, we find that the resulting displacement is small; see Appendix E.

We conclude that forces $F_2^{\text{erase}}(\mathbf{x}^{(1)})$ are always nonzero by definition of continual learning. While linear and $\xi > 1$ spring networks are too soft to protect their minima from such destabilizing forces, unstrained nonlinear $\xi \leq 1$ springs can exert strong restoring forces to counter F^{erase}, up to a threshold F_T. Forces larger than this threshold will destabilize stored states even for $\xi \leq 1$ springs; F_T thus sets the capacity, i.e., the largest number of memories that can be learned, since the total destabilizing force grows with the number of encoded states.

A. Nonlinear springs as Bayesian priors

We can see the wider applicability of nonlinear stabilization through a mathematical connection between sparse regression in statistics [34] and mechanical nonlinearities.

How does a frustrated network of nonlinear springs decide which springs should be strained at stable configurations? In Fig. 5, we consider two springs learned as part of two different configurations $\mathbf{x}^{(1)}$ (blue) and $\mathbf{x}^{(2)}$ (red). Spring a is unstrained at A while spring b is unstrained at B. The total energy of the system shown is

$$E = -\mathbf{F}^{\text{ext}}\mathbf{x} + k\sum_{q=a,b} s_q^\xi = -\mathbf{F}^{\text{ext}}\mathbf{x} + k\|\mathbf{s}\|^\xi, \qquad (3)$$

where $\|\mathbf{s}\|^\xi$ is the ξ-norm of the strain vector $\mathbf{s}$ for the red and blue springs and $\mathbf{F}^{\text{ext}}$ represents forces by other springs not shown.

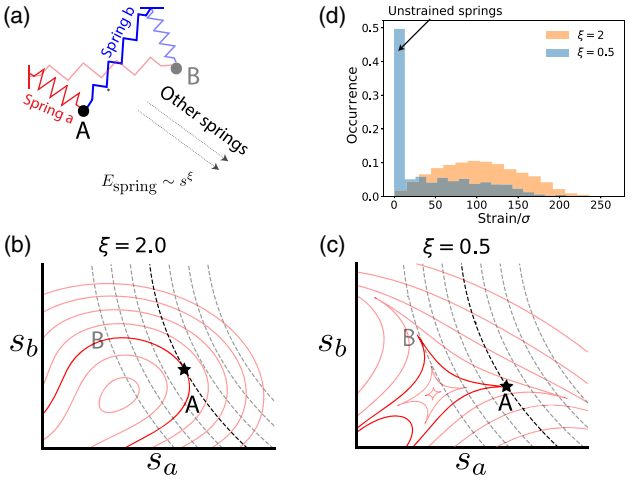

FIG. 5. Nonlinear springs act like a Bayesian prior that ensures sparse strain distributions. (a) Red spring a stabilizes the ball position at point A, while the blue spring b stabilizes the ball at position B. (b) The energy of the red and blue springs is represented by red contours, that of all other springs by black contours. The system's energy is minimized at a point where red and black contours are tangent to each other. If $\xi > 1$, the minimum is at a generic point with no special features. (c) If $\xi \leq 1$, the minimum is very likely to be at a red cusp, corresponding to a configuration in which either the red or blue springs is unstrained. (d) Typical stable states of a large $N = 100$ network have many unstrained springs if and only if $\xi \leq 1$.

In Fig. 5(b), we find that contours of constant $\|\mathbf{s}\|^{\xi}$ are smooth for $\xi > 1$. But for $\xi < 1$, the contours are cusped; at each cusp, one spring is completely unstrained while the other is strained ($\sigma = 0$ for simplicity). When combined with other springs in the network (black energy contours), the energy minima are overwhelmingly likely to be at these cusps.

In fact, this result has been established more generally in the context of sparse regression [34,35]. As an example, consider an underdetermined problem $A\mathbf{s} = \mathbf{b}$ for a vector $\mathbf{s}$. If we know *a priori* that an $\mathbf{s}$ exists which has some components that are strictly zero and others nonzero, we can find such "sparse" solutions $\mathbf{s}$ by adding a "Bayesian prior" $\|\mathbf{s}\|^{\xi} = \sum_q s_q^{\xi}$ to the least-squares loss function,

$$E = \|A\mathbf{s} - \mathbf{b}\|^2 + \|\mathbf{s}\|^{\xi}, \tag{4}$$

and then minimizing E. If $\xi \leq 1$, such a Bayesian prior $\|\mathbf{s}\|^{\xi}$ has been proven to bias the search toward solutions $\mathbf{s}$ in which some elements of $\mathbf{s}$ are strictly zero while others are nonzero (i.e., sparse solutions) [34,36].

Thus, nonlinear spring networks with $\xi \leq 1$ are predicted to be winner-take-all networks; in stable configurations, one subset of springs is completely unstrained while others are strained. In $\xi > 1$ networks, strain is democratically shared and no springs are unstrained.

To test this analogy in larger elastic networks, we let a $N = 100$ particle network learn two distinct states, and measured the strain in each spring after relaxing to one of the states [Fig. 5(d)]. For nonlinear springs $\xi \leq 1$, we find a bimodal strain distribution—half of the springs are considerably strained, while the other half are at (approximately) zero strain. This result is in stark contrast to the simultaneously stabilized minima with linear springs $\xi = 2$, for which all springs are strained. Thus nonlinear springs can be seen as sparse Bayesian priors.

B. Pattern recognition

Finally, we ask whether our learned network with large robust attractors around the learned states can perform pattern recognition. To do this, we turn to the MNIST handwritten digits database [37] and attempt to teach an elastic network to recognize the digits "0" and "1" from examples of these digits (Fig. 6).

We trained the fully connected ($R \to \infty$) elastic network with 5000 samples of the digits 0 and 1 each from the MNIST database in the following way: each 400 pixel image was interpreted as a 1D configuration of 400 particles by interpreting each pixel's gray scale value as a particle's position in the interval [0, 1]. The particles in such a state are connected by elastic rods according to the learning rule in Eq. (1). For $\xi < 1$, we find that the training generally creates two distinct large attractors corresponding to an idealized 0 and 1, respectively [Fig. 6(d)].

We then test the network by using novel unseen examples of 0's and 1's from MNIST as initial conditions

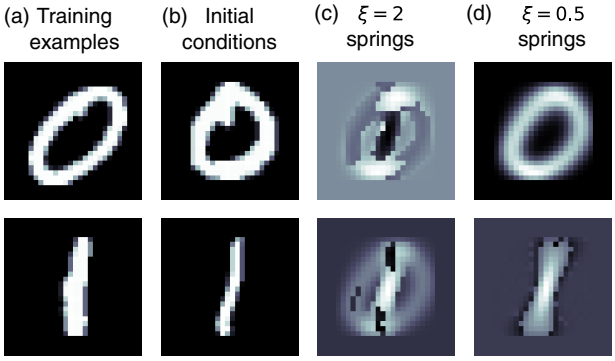

FIG. 6. Elastic networks learn to recognize handwritten digits. (a) Images representing two particle configurations that we wish to stabilize (adapted from MNIST). The 400 pixel gray scale values in each image are interpreted as positions of 400 particles in one dimension. We learned a nonlinear spring network using 5000 randomly drawn examples of 0's and 1's each. (b) Learned networks are then tested by initializing at configurations corresponding to new unseen examples of "0" and "1." (c) Linear networks fail to learn stereotyped states; initializing at each test example results in n unique uninterpretable states. (d) In contrast, nonlinear networks learn two stereotyped states corresponding to 0 and 1 that are reliably retrieved in response to unseen examples of 0 and 1 from the MNIST database ($\xi = 0.5$, $\sigma = 5 \times 10^{-3} L$).

for the particles. While these test images are not identical to any particular 0 or 1 used in training, the elastic network still retrieves the correct stored 0 or 1 state. Thus the nonlinear $\xi < 1$ elastic network learns states 0 and 1 with sufficiently large attractors to accommodate the typical handwriting variations seen in the MNIST database.

IV. EXPERIMENTAL REALIZATIONS

In this work we propose growing networks as a potential mechanical system that can continually learn stable states. The essential requirement for such systems is the ability to grow or adapt bonds up to a particular length when nodes are held in position corresponding to one of the stored states, as shown in Eq. (1). Formation of connections between node points occurs naturally in cellular microtubule networks, where filamentous networks grow to form the mitotic spindle, allowing cells to divide with their daughter cells sharing the chromosomes equally [18]. A similar synthetic system where our framework can be tested is DNA nanotubes [20]. Much as in our model, in the experiments of Ref. [20], DNA nanotubes spontaneously grow between different "nodes" (DNA origami seeds) that can be placed at desired locations on a surface. The probability of connecting up two nodes by a self-assembled DNA nanotube drops off sharply as a function of the distance between the nodes and the time allowed for assembly [Fig. 7(a)]; such a system effectively implements our learning rule Eq. (1) with $f(r)$ chosen as a step function. Further, using multiheaded DNA origami seeds, one can allow for multiple nanotubes to grow from each node, including multiple tubes between a pair of links, creating a network with links of varying strength. In this way, the highly controllable nature of DNA can implement geometry-dependent growth rules that underlie this work on the nanometer to micron scale.

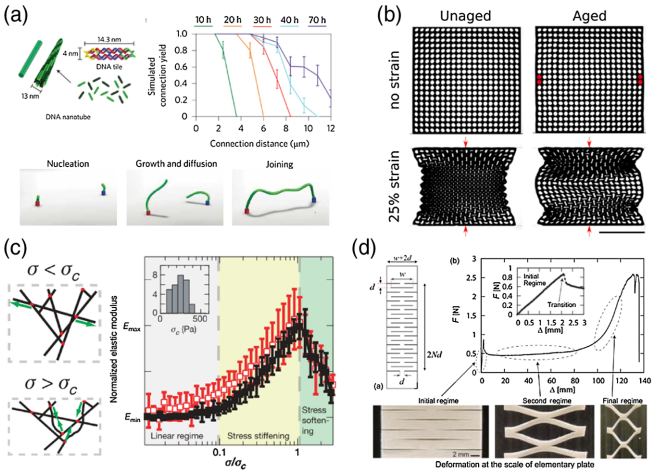

FIG. 7. Experimental networks exhibiting components of learning and nonlinear elasticity. (a) Synthetic DNA origami nanotubes forming connections up to certain distance, as our chosen $f(r)$ (adapted from Ref. [20]). (b) EVA foam-based network changes nonlinear elastic properties when aged under strain (adapted from Ref. [38]). (c) Reversible stress softening in actin networks, a natural form of elastic nonlinearity in grown networks (adapted from Ref. [39]). (d) Kirigami structure with $\xi \approx 1$ nonlinear elasticity (adapted from Ref. [28]).

Our framework can also be implemented on much larger length scales using recent experimental demonstrations of plasticity in EVA (Ethylene-vinyl acetate) foam [38]. Here, learning involves continually changing the stiffness of existing bonds, rather than growing new bonds every time. Such a method was recently explored using a "holey sheet" made of EVA foam. These sheets can be trained to have a very different response by directed aging of the network under strain [Fig. 7(b)]. During the aging process, different bond stiffnesses are modified to different extents and the geometry is altered as the EVA foam remodels. Using such directed aging methods, the nonlinear elasticity of the sheet is modified and it may be possible to train the sheet to be multistable, with energy minima at specific strains. These holey sheets can then be used as individual springs in our network, instead of the growing springs studied here, with an overall similar multistability obtained at the network level.

In this work we find that given such adaptive materials, continual learning of stable states is possible if the interactions between the nodes is sufficiently nonlinear. Though most materials respond linearly to small strains, nonlinear elasticity is quite common when larger strains are considered [40]. Nonlinear elastic responses are typically strain hardening (restoring force that grows superlinearly with strain, $\xi > 2$) or irreversible strain softening (e.g., material failure). However, mechanical systems that support reversible strain softening do exist, with various effective values of ξ. A synthetic system that clearly shows the type of elasticity we require for continual learning is the kirigami spring [28], where a nonlinear elastic regime (with $\xi \sim 1$) is observed when the kirigami structure buckles at a finite strain [Fig. 7(d)]. If a growing network of protein filaments could be engineered with such topology, it could be ideal for testing our framework for continual learning.

Actin networks are known to soften at large strains [39]. At moderate strains the network hardens due to hardening of individual fibers, but at larger strains fiber groups may buckle, diminishing the restoring force [Fig. 7(c)]. In addition, actin networks can also grow between target points, as done naturally in focal adhesion, where actin networks connect to focal complexes to facilitate cell motility [41]. Such networks can be grown to stabilize two distinct geometrical states, each of which is supported by just a part of the network. When placed in one state, the actin bundles corresponding to the second state may weaken to the point where only the bundles supporting the current state are important. Then, if the network is transferred to the other state, the bundles associated with it can heal and stabilize the second state.

V. DISCUSSION

In this work we demonstrated a continual learning framework for creating multistable elastic networks. We found that continually learning novel states without overwriting existing ones requires a specific nonlinear elasticity $\xi \leq 1$. The specific learning model we study relies on spontaneous growth of stabilizing rods between nearby nodes, a behavior displayed by microtubules [18], DNA nanotubes [19], and other such seeded self-assembling tubes [42–44]. We believe that actomyosin networks [39] may be an ideal system to test our ideas on continual learning, as they exhibit nonlinear elasticity of a similar form to the one studied here.

The nonlinearity ξ plays a unique role as a material design parameter. Most material parameters (e.g., l_i, k_i of springs) encode information about desired states. The power parameter ξ encodes an assumption on how information about desired states is distributed among parameters l_i, k_i of different springs. Continual learning requires localization of information about each state to a subset of all springs. Hence, stabilizing learned states requires $\xi < 1$, establishing states in which some springs are fully relaxed even if others are highly strained; i.e., the strain profile is sparse. In this way, the nonlinearity ξ is mathematically analogous to Bayesian priors in statistical regression that encode assumptions about the sparse nature of solutions. However, the elastic network here goes beyond the classic sparsity problem [Eq. (4)]; the network has 2D spatial geometry absent in Eq. (4) and is more closely related to (unsolved) sparse reconstruction of 2D maps from pairwise distances between cities [45]. Consequently, we can explore how physical parameters with no analog in Eq. (4), such as the maximum range of learned interactions R [Fig. 3(d)] and spatial correlations between stored states, affect the optimal nonlinearity ξ (Appendix D).

The frameworks of simultaneous stabilization and continual learning have complementary strengths, as seen before in neural networks and spin glasses. For example, Hopfield [23] introduced neural networks that can continually learn arbitrary novel memories using a biologically plausible "Hebbian" learning rule. Gardner [24] showed that the same model has a higher memory capacity if we assume an optimal network construction. However, Gardner's network construction requires that all desired memories are known—and must be remade from scratch to include new memories.

Similarly, in materials, standard approaches might be sufficient if all desired states are known beforehand and given unlimited computation power, as such frameworks allow optimization over all network parameters. In contrast, continual learning is a physically constrained exploration of the same parameters. However, such constrained exploration can be superior when the desired behaviors are not known *a priori* and revealed only during use of the material itself.

Recent work has explored *in situ* supervised learning [46] in mechanical systems. Our work here is more akin to unsupervised learning (e.g., Hopfield models [23]); we leave continual learning in the supervised context and potential relationship to mechanical nonlinearities to future work. We hope the mechanical model studied here will stimulate further work on realistic learning rules that allow materials to acquire new functionalities on the fly.

ACKNOWLEDGMENTS

We thank Miranda Cerfon-Holmes, Sidney Nagel, and Lenka Zdeborova for insightful discussions and Vincenzo Vitelli for a careful reading of the manuscript. This research was supported in part by the National Science Foundation under Grant No. NSF PHY-1748958. We acknowledge NSF-MRSEC (Materials Research Science and Engineering Centers) 1420709 for funding and the University of Chicago Research Computing Center for computing resources.

APPENDIX A: SIMULTANEOUS STABILIZATION OF STATES WITH LINEAR AND NONLINEAR SPRINGS

As a simple model for weakly strained elastic materials, linear (Hookean) springs are often used for theoretical constructions of elastic networks. Each of the two nodes connected by a linear spring of stiffness k and rest length l, and separated by distance r, feels a force $|\mathbf{F}| = k|r - l|$. The energy associated with the straining of the spring is $E = \frac{1}{2}k(r-l)^2$.

Suppose we construct a network with N nodes embedded in d-dimensional space. Each 2 nodes (located at $\mathbf{x}_i$, $\mathbf{x}_j$) are connected by a linear spring of stiffness k_{ij} and rest length l_{ij}. The energy of the elastic network is

$$E(\{\mathbf{x}\}) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=i+1}^{N} k_{ij}(r_{ij} - l_{ij})^2, \tag{A1}$$

where $r_{ij} \equiv \|\mathbf{x}_i - \mathbf{x}_j\|$ are the distances between nodes. The stable configurations (minima) of this energy function are found by equating the gradient of Eq. (A1) with respect to node positions to zero:

$$0 = \partial_{\mathbf{x}_a} E = \sum_{i=1}^{N}\sum_{j=i+1}^{N} k_{ij}(r_{ij} - l_{ij})\frac{\partial r_{ij}}{\partial \mathbf{x}_a}. \tag{A2}$$

This procedure gives Nd equations that have to be satisfied simultaneously for the Nd node coordinates. Note that Eq. (A2) is not linear in node coordinates, as the distances in dimension d are computed by $r_{ij} = \sqrt{\sum_d (x_{i,d} - x_{j,d})^2}$ (manifestly nonlinear in x_i for $d > 1$, but even for $d = 1 \rightarrow r_{ij} = |x_i - x_j|$). Because of the nonlinear relation of r_{ij} to x_i, x_j, multiple solutions

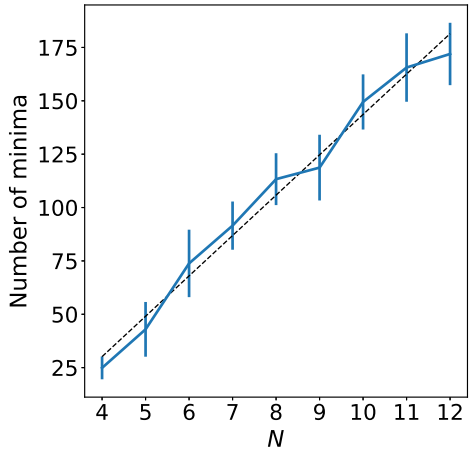

FIG. 8. Number of stable configurations in a network of linear springs grows linearly with the size of the system.

$\{\mathbf{x}^\star\}$ can satisfy Eq. (A2). Even though one still needs to check the second derivative at the proposed configuration $\{\mathbf{x}^\star\}$ to test if it is a stable minimum, in practice we find that there indeed exist multiple stable points for two-dimensional embeddings. Simulating small systems with up to 12 nodes in 2D, we find that the number of minima scales linearly with node number (Fig. 8).

These multiple minima in the potential energy landscape, if moved around, could be utilized to encode the desired stable configurations. This is possible by careful choice of the stiffness values k_{ij} and rest lengths l_{ij} of all springs. Note that even though Eq. (A2) is nonlinear in node positions $\{\mathbf{x}\}$, it is linear in both k_{ij} and $a_{ij} \equiv k_{ij} l_{ij}$. Suppose we want to solve the system of equations (A2) for M different node configurations denoted by $\{\mathbf{x}\}^m$, giving rise to distance matrices r_{ij}^m. A solution to such linear systems of equations can generally be found if the number of equations (NdM) is less than the number of variables ($N^2 d$). To simultaneously stabilize multiple desired states, we thus numerically solve Eq. (A2) for all the desired configurations $\{\mathbf{x}\}^m$ to get the values of k_{ij}, l_{ij}, and then check that the obtained elastic network is indeed stable in all of these configurations. Similar strategies for multistability in elastic networks were recently studied [13] from the point of view of spring length constraints.

The particular algorithm discussed above is only relevant for linear springs with $\xi = 2$. Still, a simultaneous stabilization protocol for spring-node systems with any value of nonlinearity ξ is possible. With nonlinear springs the force balance of Eq. (A2) becomes

$$0 = \partial_{\mathbf{x}_a} E \sim \sum_{i=1}^{N} \sum_{j=i+1}^{N} k_{ij} (r_{ij} - l_{ij})^{\xi - 1} \frac{\partial r_{ij}}{\partial \mathbf{x}_a}, \quad \text{(A3)}$$

which is unfortunately nonlinear in the rest lengths l_{ij}. In similar spirit to the above algorithm, we minimize the sum squared of all NdM equations due to the set of Eq. (A3) over the spring parameters k_{ij}, l_{ij}. If minimization succeeds in finding perfect (zero) solutions, it gives sets k_{ij}, l_{ij} for which the nodes feel very little force in all of the M stable states. We can then numerically check whether these states are stable.

The capacity of such simultaneously stabilized networks to store multiple stable states M_C is expected to scale linearly with system size (number of nodes N). This idea arises as stabilizing M states requires the simultaneous satisfaction of NdM constraints using $N^2 d$ parameters as discussed above. These two numbers match for a critical number of states $M_C \sim N$, and for $M > M_C$ no solution exists in general. Unfortunately, this prediction is difficult to corroborate numerically due to the computationally NP-complete nature of such design problems.

APPENDIX B: ENERGY MODEL FOR NONLINEAR SPRINGS

To enable the learning paradigm to store multiple stable states in an elastic network, one needs to utilize nonlinear springs with certain properties. Most importantly, if a spring is to hold information about one configuration associated to it, the spring should apply a strong force only when the system is close to its associated configuration. One simple way to parametrize such forcing is to use a spring whose force when pulled away from the preferred length is $F \sim (r - \ell)^{\xi - 1}$. Clearly, if one chooses $\xi = 2$, the limit of linear springs is obtained once more, where the force gets stronger the further the spring is strained.

If one chooses $0 < \xi < 1$, the spring's response weakens as it is strained. Unfortunately, such springs are nonphysically singular for $r = \ell$. One way to counter this singularity is to introduce a linear "core" spring, with some length scale σ, such that the spring behaves like a linear spring for $|r - \ell| < \sigma$, and nonlinearly otherwise. If we define a nondimensional strain $u \equiv (r - \ell)\sigma^{-1}$, the energy of such a spring can be written as

$$E(u) = \frac{1}{2} k \sigma^{\xi} \frac{u^2}{(1 + u^2)^{1 - (1/2)\xi}}, \quad \text{(B1)}$$

with r the spring length, k stiffness, and l, σ the rest length and core size, respectively. The prefactor σ^{ξ} is chosen so that the long-range forces $u \to \infty$ are independent of the core size σ, and that the $\xi = 2$ limit is the desired linear spring. In this model, spring nonlinearity is controlled by the exponent ξ, defined in a way to recapitulate the behavior of regularizers in optimization problems. A choice of $\xi = 2$ gives rise to linear springs, akin to ridge regularization, while $\xi = 1$ gives long-range constant forces $E \sim u$, similar to LASSO (least absolute shrinkage and selection operator; also Lasso or lasso) regularization. The extreme limit $\xi = 0$ defines springs whose energy is a Lorentzian. Outside the core region, such springs exert

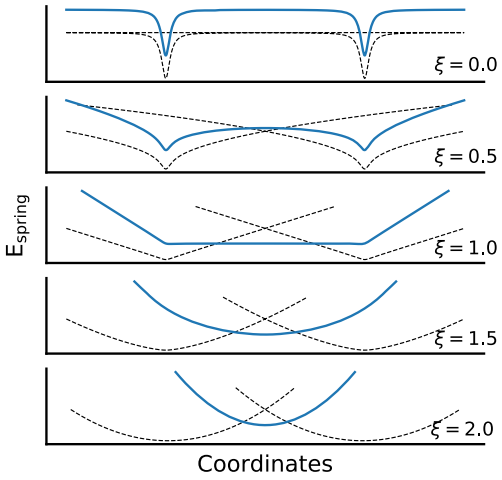

FIG. 9. The sum energy of two springs goes through a transition at $\xi = 1$. The energy minimum of each spring is preserved for $\xi < 1$, while these minima are overwritten for $\xi > 1$. In essence, the information on minima of $\xi < 1$ springs is stored with each individual spring. (Black dotted lines correspond to the individual potentials of two nonlinear springs with given ξ, bold blue lines show the sum of the two potentials, shifted up for clarity).

forces that diminish quickly as $F \sim u^{-1}$. In general, the force due to the nonlinear springs is

$$F(u) = k\sigma^{\xi-1} u \frac{1 + \frac{1}{2}\xi u^2}{(1+u^2)^{2-(1/2)\xi}}. \tag{B2}$$

The crucial property of this family of spring potentials is the force behavior at large strains, far beyond the core $u \gg 1$. At large strains the force applied by the springs is $F \sim u^{\xi-1}$, a form which goes through an important transition at $\xi = 1$. For springs with $\xi > 1$, the restoring force grows with strain, while for $\xi < 1$ the force diminishes. This transition causes an important change of behavior when such spring potentials are summed together, as shown in Fig. 9. The minima of individual springs are preserved for $\xi < 1$, while these minima are overwritten for springs with $\xi > 1$. We conclude that only springs with $\xi < 1$ (or more generally, springs whose force diminishes with range) enable continual learning. In continual learning, we would like the information about each stored state to be localized to a subset of springs, and that adding more springs for new states does not overwrite the previously stored information. Figure 9 clarifies that springs whose force grows with strain are unfit for this purpose.

APPENDIX C: NUMERICAL EXPLORATION OF MECHANICAL NETWORKS

Testing predictions about learning elastic networks requires the numerical construction of such networks and the ability to explore their potential energy landscape. This Appendix describes some of the technical aspects involved in simulating these networks and deducing their properties. The codes to produce and study the elastic networks are implemented in PYTHON and available upon request.

1. Network construction

The elastic networks simulated for this work consist of N nodes embedded in a 2D box of size 1×1. For each desired system configuration (stored state), node positions are sampled uniformly at random within the boundary of the box. Each multistable system of this type with M states is thus described by $M \times N \times 2$ positions in the range [0, 1]. Springs are attached between pairs of nodes according to the paradigm studied (simultaneous stabilization, continual learning).

For the simultaneous stabilization model, we fully connect all pairs of nodes in the system with linear springs ($\xi = 2$). These springs are chosen to take into account all of the desired states simultaneously. The choice of springs (stiffness and rest length values) is made by solving the set of equations (A2) in Appendix A. Construction of fully connected, simultaneously stabilized networks with nonlinear springs ($\xi \neq 2$) is facilitated by optimizing forces at the desired stored states (Appendix A).

Systems with learned states are constructed by attaching a set of springs between pairs of nodes for each stored state. We generally do not fully connect the nodes, instead opting to connect a spring between nodes within a certain chosen distance R. All springs in this paradigm have the same spring stiffness k, core size σ, and nonlinearity parameter ξ. The springs only differ in their rest length, chosen so that the springs are relaxed in their respective state. Thus, learning is "easy" in the sense that no computation is required to choose the new set of springs in new stored states. This suggests learning can be performed by a rather simple, physically passive system, whose time evolution depends only on its current configuration.

2. Estimation of attractor size and barrier height

When $M > 1$ states are encoded into a network, it is of immediate interest to check whether these states are stable at all. We define a stable state $\vec{X}^{(m)}$ ($N \times 2$ spatial vector) by the following requirement: when the system is released from $\vec{X}^{(m)}$ and allowed to relax to a nearby stable minimum of the potential energy landscape, the relaxed configuration $\vec{X}_*^{(m)}$ is close in configuration space to $\vec{X}^{(m)}$. We consider states to be preserved if the average displacement per degree of freedom after relaxation is much smaller than the size of the box. The potential core size σ is used as this stability cutoff $[(\|\vec{X}_*^{(m)} - \vec{X}^{(m)}\|)/2N] < \sigma$, where the typical core size is $\sim 1\%$ of the box size. If the different encoded states pass this test, we say that the states are stable, and the encoding was successful. See Appendix D for more details on the stability of stored states.

In an effort to find optimal schemes for storing stable states in elastic networks, basic stability does not suffice, and we require additional measures of merit. A natural approach is to study the attractor basins of the encoded states, specifically their spatial extent and the energetic barriers surrounding them. The larger the attractor basin, the configuration can more reliably be retrieved when the system is released farther away from its minimum. High energy barriers surrounding the state basins improve their stability when the system is subjected to finite temperatures or other types of noise.

Unfortunately both attractor size and energy barrier are nonlocal properties of the attractor, requiring many high-dimensional measurements away from the stable state. Rather than exhaustively studying the attractor basin shape and height, we employ a procedure as follows. At the stable state, choose a random direction and take the system a small amount in that direction. Relax the system from the new position and verify whether it relaxed into the same stable state. If so, repeat the last step, but take the system slightly farther away in the same direction as before. Repeat these steps until the system no longer relaxes to the initial state, but instead reaches another stable point of the landscape. Measuring the distance required to move the system in order to escape the attractor, and the energy at that distance, furnishes an estimate of both the attractor size and the energy barriers around it. We repeat the above process to average the results over many different random directions in configuration space.

An important correction is needed for the above estimation, in particular for the flatter spring potentials $\xi \ll 1$. Attractors arising from these potentials tend to be very flat far from the core region σ surrounding each stored state. Although flat regions mathematically belong to some attractor basin, releasing the system in these regions will require long relaxation times, and relaxation dynamics are highly unstable to external noise. We therefore define a "useful" attractor, such that the gradient that leads relaxation toward the stable point is large enough. In practice, we cut off the attractor defined by the previous algorithm when the relaxation force is smaller than a fraction ($\sim$0.5) of the typical force within the core distance σ. The inclusion of this force (gradient) requirement gives rise to an optimal nonlinearity value $0 < \xi < 1$ for learned states.

APPENDIX D: STABILITY OF LEARNED STATES

We established the necessity of nonlinear springs as a means of continually learning multiple stable states in an elastic network. In this Appendix, we discuss some limitations of this idea, such as the finite capacity of node-spring networks, and the effect of connectivity within a state and correlations between states on the quality of learning.

1. Storing capacity

Nonlinear spring networks (with $\xi < 1$) can stabilize multiple states through sparsity—springs associated with a certain state dominate the response of the network when it is situated close to that state. Springs associated with other states are highly stretched, yet apply small forces that further diminish at high strains. Still, force contributions of springs unrelated to the desired state are finite and act to destabilize that state.

The learned networks studied in this work exhibit destabilization of learned states due to the effect of springs associated with other stored states. Figure 10(a) shows a typical scenario observed in these networks, where a desired state is stable when the overall number of learned states is low. Then, an abrupt threshold (capacity) is passed after which the state destabilizes completely and the system relaxes into a configuration that looks completely different from the desired stored state. Generically, all learned states fail in this way at a similar capacity value [Fig. 10(b)]. This capacity is well defined and observed to depend on the parameters of the system, such as size N and nonlinearity ξ.

Let us now argue for a scaling relation of the storing capacity. Suppose a system of N nodes is used to learn $M + 1$ states. In configurations close to state 1, N springs will apply a stabilizing force F_S, while the rest $N \times M$ springs will act to destabilize the state with force F_{DS}. All stabilizing springs provide a force in the same stabilizing direction such that $F \sim N$. If we assume the $N \times M$ destabilizing forces due to unrelated springs are randomly oriented and similar in magnitude, the total destabilizing force would behave like a random walk and have a magnitude $F_{\mathrm{DS}} \sim \sqrt{N \times M}$. The capacity of the system is reached when the magnitude of the destabilizing force is equal to that of the stabilizing force, so that

$$F_{\mathrm{DS}}(M_C) \sim F_S \rightarrow M_C \sim N. \tag{D1}$$

The capacity of a learning network is expected to scale linearly with system size, similarly to other Hopfield-inspired learning models [23]. This prediction was tested in networks of sizes $N = 6$–26 and for several values of the nonlinearity ξ. Results shown in Fig. 10(c) are consistent with the linear scaling suggested above. Theoretical arguments of a similar nature suggest another scaling relation $M_C \sim \exp(-\xi)$, also in agreement with numerical data. However, we regard the capacity dependence on nonlinearity to be of lesser interest, as other metrics for quality of encoding (barrier height and attractor size) are more important for the robustness of learning.

2. Connectivity of nodes

It is well known that the rigidity of elastic networks strongly depends on node coordination—the number of springs connected to the different nodes. Rigid networks are characterized by coordination numbers exceeding the

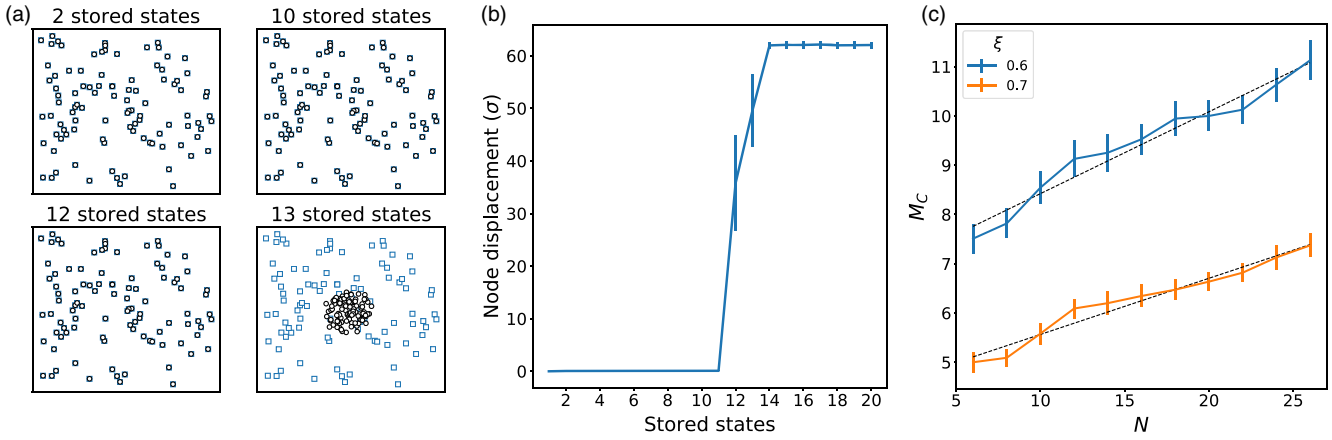

FIG. 10. Programming stored states using the learning paradigm exhibits finite capacity. (a) Each stored state is affected by springs associated with other states. Initially the new springs have a small effect and the state remains a stable attractor. However, eventually states destabilize due to the forces exerted by the other stored states (blue squares denote a certain stored state, black circles show the nearby stable configuration). This state fails when 13 states are simultaneously encoded ($N = 100$, $\xi = 0.6$, $\sigma = 10^{-2}L$). (b) When node displacement is averaged over stored states, we observe a sharp failure of all states at a specific load, defined as the *capacity* (12–13 states in this case). (c) Capacity scales linearly with system size N.

Maxwell condition [47]. A stable state of the overconstrained network can be understood as a minimum point of the energy landscape constructed of the spring potentials. Further increasing the coordination of nodes—or their connectivity to other nodes—results in stable states surrounded by higher energy barriers.

This argument suggests an intriguing possibility, that increasing connectivity in a learned network may improve the stability and quality of the state storage. Such an outcome is possible as the act of adding more nonlinear ($\xi < 1$) springs associated with a certain stored state is not expected to significantly alter the state itself, since the rest lengths are chosen to stabilize this state. On the other hand, the extra springs may increase the height of energy barriers surrounding the state, making it more stable against temperature and noise. Furthermore, increasing connectivity may also enlarge the attractor regions of stored states, as the extra constraints induced by the new springs may suppress "distractor" states (spurious energy minima due to partial satisfaction of the frustrated interactions).

In the context of our learning paradigm, connectivity is controlled by the effective radius of rod growth R. If states are constructed by randomly placing N nodes in a d-dimensional square box of length L, it is easy to see that the average connectivity scales as $\langle Z \rangle \sim NR^d$ while $R \ll L$. We use $N = 100$, $\xi < 1$ networks to test the effect of node connectivity on the attractor size of stored states. Results presented in Figs. 11(a)–11(c) verify that the quality of state storage, as measured by the attractor size of states, improves with their connectivity.

3. State similarity

In most of this work we considered stored states that are completely uncorrelated between themselves; i.e., the position of a node in each stored state is independent of its position in other states. In practice, it might be easier to conceive of elastic networks whose different stable states are not too different from one another, in which neighboring nodes in one configuration will remain neighbors in other configurations. Furthermore, some applications (e.g., classification of similar objects) may require different stored states to be correlated to differing extents. In general, encoding correlated (i.e., similar) states is expected to negatively affect the stability of these states and their quality (as measured by attractor properties as size and barrier heights).

To test the impact of similarity between states, we embedded a $N = 100$ network with states in which the average displacement of nodes in successive states was controlled. Figures 11(d)–11(f) show that the larger the difference between states, the larger their respective attractor sizes. In addition, larger differences between states allows their stabilization at higher ξ values, which is expected to improve the heights of energy barriers surrounding them and further suppress distractor states. Still, we show that it is possible to encode multiple states in elastic networks, even when the average difference between stored states is a small multiple of σ (the potential core size, within which states are indistinguishable).

APPENDIX E: CONTINUAL LEARNING REQUIRES NONLINEAR, UNSTRAINED SPRINGS

In the main text we laid out arguments for the necessity of nonlinear $\xi < 1$ springs for continual learning. Here, we expand on these considerations, specifically showing that continual learning always results in deformation of learned states. Thus, nonlinear $\xi < 1$ springs, causing such

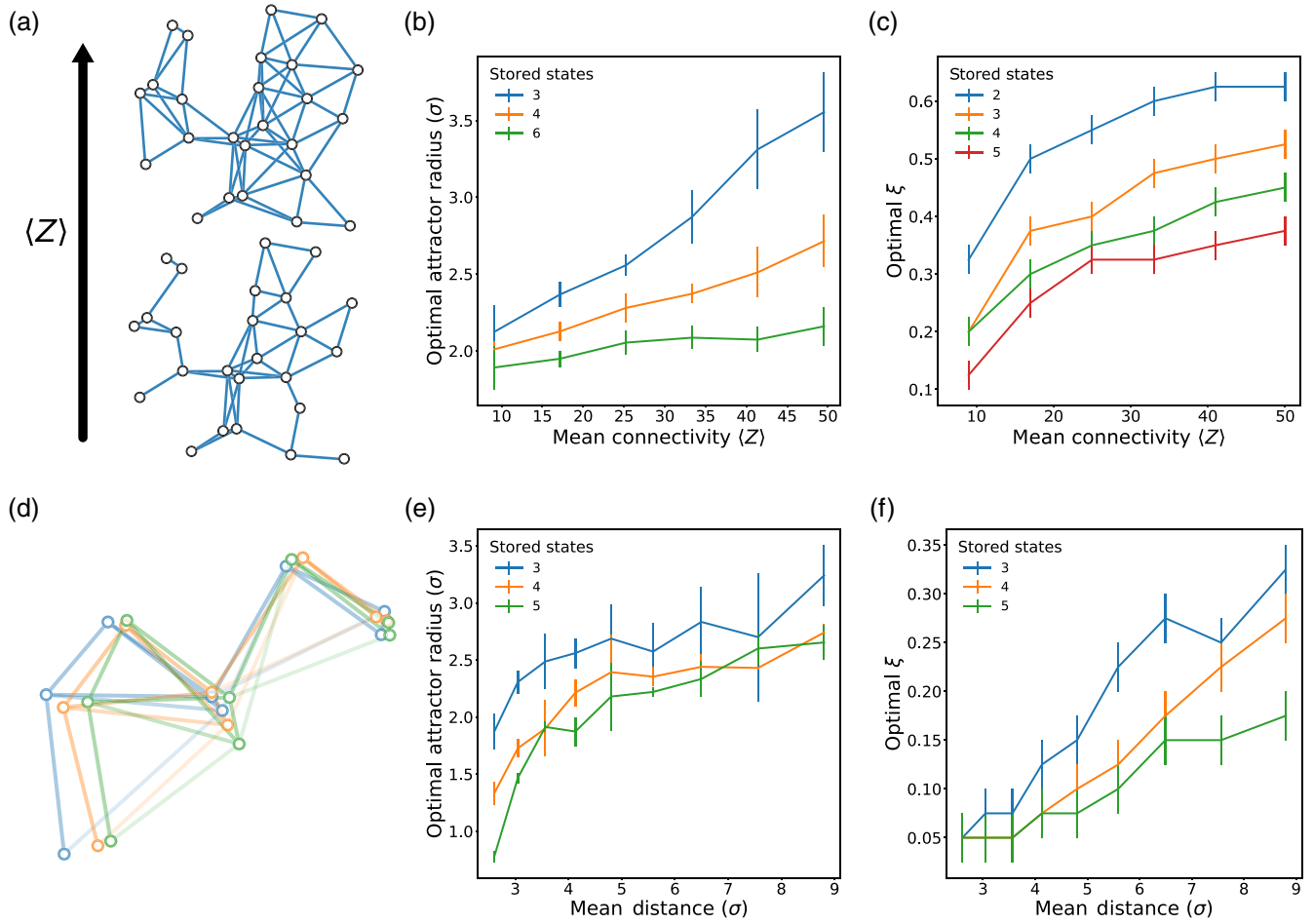

FIG. 11. Effects of node connectivity and state similarity on the quality of encoding the stored states. (a)–(c) Connectivity between nodes $\langle Z\rangle$ increases with the effective connection radius R. We find that the more internally connected a state is, the larger its attractor size, and higher the optimal value of the nonlinearity parameter ξ ($N = 100$, $\sigma = 5 \times 10^{-3}L$). (d)–(f) Trying to store similar states is more difficult than random states. When the mean distance between nodes in successive states is small, attractor's basin is also small, and successful encoding requires small values of ξ and flat potentials ($N = 100$, $\sigma = 5 \times 10^{-3}L$). Results averaged over six simulations of random continual learning networks.

deformations to be small, are essential for continual learning. We also discuss why the nonlinear springs need to be unstrained in their respective states to stabilize them.

1. Continual learning always deforms learned states

Suppose we have M sets of linear $\xi = 2$ springs, chosen such that a state $\mathbf{x}^{(1)}$ is stable. These spring sets are defined so that one such set can be modified to accommodate a single new stable state in the system. In this setting, we can write a force balance equation similar to Eq. (A2):

$$0 = \partial_{\mathbf{x}_a} E(\mathbf{x}^{(1)}) = \sum_{m=1}^{M}\sum_{i=1}^{N}\sum_{j=i+1}^{N} k_{mij}(r_{mij}(\mathbf{x}^{(1)}) - l_{mij})\frac{\partial r_{mij}(\mathbf{x}^{(1)})}{\partial \mathbf{x}_a}. \quad \text{(E1)}$$

Continually learning a new state $\mathbf{x}^{(2)}$ is only allowed through the modification of the springs of set no. 2. Suppose we devise a learning rule that after a short application, infinitesimally changes the properties of spring set no. 2:

$$k_{2ij} \to k_{2ij} + \Delta k_{2ij},$$
$$l_{2ij} \to l_{2ij} + \Delta l_{2ij}.$$

Reevaluating the force equation, we note that state $\mathbf{x}^{(1)}$ remains stable only if the force balance is maintained given this spring modification:

$$0 = \partial_{\mathbf{x}_a} E(\mathbf{x}^{(1)}; k_2 + \Delta k_2, l_2 + \Delta l_2) - \partial_{\mathbf{x}_a} E(\mathbf{x}^{(1)}; k_2, l_2).$$

Maintaining first-order contributions in Δk_2, Δl_2, we obtain

$$0 = \sum_{i=1}^{N}\sum_{j=i+1}^{N}\left[\frac{\Delta k_{2ij}}{k_{2ij}} - \frac{\Delta l_{2ij}}{r_{2ij}(\mathbf{x}^{(1)}) - l_{2ij}}\right] \times (r_{2ij}(\mathbf{x}^{(1)}) - l_{mij})\frac{\partial r_{2ij}(\mathbf{x}^{(1)})}{\partial \mathbf{x}_a}. \tag{E2}$$

This is a set of linear equations in Δk_2, Δl_2. The number of variables is the number of spring parameters, which in overconstrained networks is larger than the number of equations (locations of the nodes). Finding solutions to these equations is possible only if information of state $\mathbf{x}^{(1)}$ is available. This condition prohibits any generic local learning rule that does not have information about previous states. In other words, a generic change to the parameters of the springs of set no. 2, ignorant of state no. 1, will give rise to finite residual forces in state no. 1, deforming it from the original state.

While we cannot generically solve the force balance equations for networks of $\xi \neq 2$ springs, the above arguments still hold: a generic change of the spring parameter of any set (ignorant of previous encoded states) will deform the encoded states [e.g., as in Fig. 10(a)].

2. Nonlinear, $\xi < 1$ springs support continual learning

Suppose a state $\mathbf{x}^{(1)}$ is encoded by a single unstrained spring with nonlinearity ξ and core size σ. Now, we apply an external force on the spring F^{erase} to extend it by some length δx, as shown in Fig. 4(a). The extension can be computed by considering the force balance on the spring ($u = \delta x/\sigma$):

$$F^{\text{spring}} = k\sigma^{\xi-1}u\frac{1+\frac{1}{2}\xi u^2}{(1+u^2)^{2-(1/2)\xi}} = F^{\text{erase}}.$$

If the deformation is large ($u \gg 1$), this expression simplifies to

$$F^{\text{erase}} \approx \frac{1}{2}k\xi\sigma^{\xi-1}u^{\xi-1} = \frac{1}{2}k\xi\delta x^{\xi-1}, \qquad \delta x \approx \left(\frac{2F^{\text{erase}}}{k\xi}\right)^{1/\xi-1}. \tag{E3}$$

Instead, if the deformation is small ($u \ll 1$), we may perform a Taylor expansion of the spring force around $u = 0$ to find

$$\frac{\delta x}{\sigma} \approx \frac{F^{\text{erase}}}{k}\sigma^{1-\xi}. \tag{E4}$$

If we assume that the erasure force F^{erase} is caused by a second spring with the same parameters ξ, σ, but which is highly strained $\delta x^{(2)} \equiv R \gg \sigma$, we can use Eq. (E3) to see that $F^{\text{erase}} \approx 0.5k\xi R^{\xi-1}$. Putting this force back in Eq. (E4), we finally obtain

$$\frac{\delta x}{\sigma} \approx \frac{\xi}{2}\left(\frac{R}{\sigma}\right)^{1-\xi}. \tag{E5}$$

Since we assume $R \gg \sigma$, we see that for $\xi < 1$, the deformation $\delta x/\sigma \to 0$, which is consistent with our initial assumption. On the other hand, for $\xi > 1$, the deformation diverges and our approximation fails. Fortunately, we may instead plug the erasure force in Eq. (E3), to find that for $\xi > 1$ springs, the extension is $\delta x \approx R \gg \sigma$ (as observed in Fig. 9). There is thus a transition in the spring extension due to the erasure force of another spring, at $\xi = 1$. Nonlinear springs with $\xi < 1$ will show a very small deformation $\delta x \ll \sigma$, while springs with $\xi > 1$ will exhibit $\delta x \gg \sigma$.

For finite values of the core size σ, the aforementioned considerations are valid for small enough erasure forces. If the erasure force is large, Eq. (E4) becomes inconsistent with the assumption that $\delta x/\sigma \ll 1$, and the argument fails. For $\xi < 1$ springs, this happens if the erasure force surpasses a threshold, $F^{\text{erase}} > F_T$, the largest restoring force afforded by the spring. The threshold force can be found by computing the second derivative of the spring energy and equating to zero:

$$\frac{d^2E}{du^2} \sim \frac{(1+\frac{3}{2}\xi u^2)(1+u^2) + (\xi-4)u^2(1+\frac{1}{2}\xi u^2)}{(1+u^2)^{3-0.5\xi}} = 0.$$

This equation can be solved as a function of the nonlinearity ξ to find a deformation that maximizes the restoring force:

$$u^* = \frac{\delta x^*}{\sigma} = \sqrt{\frac{\sqrt{17\xi^2 - 52\xi + 36} - 5\xi + 6}{2(\xi^2 - \xi)}} \sim o(1).$$

Note that this result does not depend on σ, so that the maximum restoring force, or threshold force, is

$$F_T \sim k\sigma^{\xi-1}. \tag{E6}$$

Thus, for unstrained nonlinear springs with $\xi < 1$ and small cores $\sigma \to 0$, the external force required for deforming an encoded state is large. Such encoded states are expected to be robust to the interference of other states and sources of noise (e.g., temperature).

3. Continual learning fails in prestrained networks

Our learning rule involves addition of unstrained nonlinear springs in every new (geometric) state to be encoded in the network. Here we show that using prestrained springs cannot facilitate continual learning. Essentially, we argue that applying external forces on prestrained springs always destabilizes the state.

Consider a spring with stiffness k and rest length ℓ. Suppose the spring is strained to an extent $r \neq \ell$ in a stable (desired) state of the network. We say this spring is

prestrained, with strain $s = r - \ell$. The magnitude of the force exerted by the spring (given nonlinearity parameter ξ and $\sigma \ll r - \ell$) in that stable state is $F = k|s|^{\xi-1}$. Now, we apply some constant external force F^{erase} parallel to the axis of the spring. The new equilibrium position of the spring will be such that $F + F^{\text{erase}} = 0$, so that the new strain is $s_* \sim (F^{\text{erase}})^{1/(\xi-1)}$. For $1 < \xi < 2$ springs, the state will deform at least in proportion to $\sqrt{F^{\text{erase}}}$. For the strongly nonlinear springs $\xi < 1$, the result is in fact much worse, and s_* tends to $\pm\infty$ for any magnitude of F^{erase}. Thus, for $\xi < 1$ springs, we conclude that the original prestrained state was an unstable equilibrium. Continual learning networks, which by definition must be stable to "external" forces due to competing states, can only be realized by using unstrained springs with $s \ll \sigma$ to encode stable states.

[1] U. Feudel, *Complex Dynamics in Multistable Systems*, Int. J. Bifurcation Chaos Appl. Sci. Eng. **18**, 1607 (2008).

[2] J. L. Silverberg, J.-H. Na, A. A. Evans, B. Liu, T. C. Hull, C. D. Santangelo, R. J. Lang, R. C. Hayward, and I. Cohen, *Origami Structures with a Critical Transition to Bistability Arising from Hidden Degrees of Freedom*, Nat. Mater. **14**, 389 (2015).

[3] S. Waitukaitis, R. Menaut, B. G.-G. Chen, and M. van Hecke, *Origami Multistability: From Single Vertices to Metasheets*, Phys. Rev. Lett. **114**, 055503 (2015).

[4] J. T. B. Overvelde, T. A. De Jong, Y. Shevchenko, S. A. Becerra, G. M. Whitesides, J. C. Weaver, C. Hoberman, and K. Bertoldi, *A Three-Dimensional Actuated Origami-Inspired Transformable Metamaterial with Multiple Degrees of Freedom*, Nat. Commun. **7**, 10929 (2016).

[5] S. Shan, S. H. Kang, J. R. Raney, P. Wang, L. Fang, F. Candido, J. A. Lewis, and K. Bertoldi, *Multistable Architected Materials for Trapping Elastic Strain Energy*, Adv. Mater. **27**, 4296 (2015).

[6] K. Bertoldi, V. Vitelli, J. Christensen, and M. van Hecke, *Flexible Mechanical Metamaterials*, Nat. Rev. Mater. **2**, 17066 (2017).

[7] G. Steinbach, D. Nissen, M. Albrecht, E. V. Novak, P. A. Sánchez, S. S. Kantorovich, S. Gemming, and A. Erbe, *Bistable Self-Assembly in Homogeneous Colloidal Systems for Flexible Modular Architectures*, Soft Matter **12**, 2737 (2016).

[8] L. Wu, X. Xi, B. Li, and J. Zhou, *Multi-Stable Mechanical Structural Materials*, Adv. Eng. Mater. **20**, 1700599 (2018).

[9] Y. Yang, M. A. Dias, and D. P. Holmes, *Multistable Kirigami for Tunable Architected Materials*, Phys. Rev. Mater. **2**, 110601 (2018).

[10] K. Che, C. Yuan, J. Wu, H. J. Qi, and J. Meaud, *Three-Dimensional-Printed Multistable Mechanical Metamaterials with a Deterministic Deformation Sequence*, J. Appl. Mech. **84**, 011004 (2017).

[11] H. Yang and L. Ma, *Multi-Stable Mechanical Metamaterials with Shape-Reconfiguration and Zero Poisson's Ratio*, Mater. Des. **152**, 181 (2018).

[12] H. Fu *et al.*, *Morphable 3D Mesostructures and Microelectronic Devices by Multistable Buckling Mechanics*, Nat. Mater. **17**, 268 (2018).

[13] J. Z. Kim, Z. Lu, S. H. Strogatz, and D. S. Bassett, *Conformational Control of Mechanical Networks*, Nat. Phys. **15**, 714 (2019).

[14] H. Sayama, I. Pestov, J. Schmidt, B. J. Bush, C. Wong, J. Yamanoi, and T. Gross, *Modeling Complex Systems with Adaptive Networks*, Comput. Math. Appl. **65**, 1645 (2013).

[15] D. Hu and D. Cai, *Adaptation and Optimization of Biological Transport Networks*, Phys. Rev. Lett. **111**, 138701 (2013).

[16] J. Gräwer, C. D. Modes, M. O. Magnasco, and E. Katifori, *Structural Self-Assembly and Avalanchelike Dynamics in Locally Adaptive Networks*, Phys. Rev. E **92**, 012801 (2015).

[17] S. L. Gupton and C. M. Waterman-Storer, *Spatiotemporal Feedback between Actomyosin and Focal-Adhesion Systems Optimizes Rapid Cell Migration*, Cell **125**, 1361 (2006).

[18] H. Hess and J. L. Ross, *Non-Equilibrium Assembly of Microtubules: From Molecules to Autonomous Chemical Robots*, Chem. Soc. Rev. **46**, 5570 (2017).

[19] A. M. Mohammed and R. Schulman, *Directing Self-Assembly of DNA Nanotubes Using Programmable Seeds*, Nano Lett. **13**, 4006 (2013).

[20] A. M. Mohammed, P. Šulc, J. Zenk, and R. Schulman, *Self-Assembling DNA Nanotubes to Connect Molecular Landmarks*, Nat. Nanotechnol. **12**, 312 (2017).

[21] M. Rumpler, A. Woesz, J. W. C. Dunlop, J. T. van Dongen, and P. Fratzl, *The Effect of Geometry on Three-Dimensional Tissue Growth*, J. R. Soc. Interface **5**, 1173 (2008).

[22] A. Tero, R. Kobayashi, and T. Nakagaki, *Physarum Solver: A Biologically Inspired Method of Road-Network Navigation*, Physica (Amsterdam) **363A**, 115 (2006).

[23] J. J. Hopfield, *Neural Networks and Physical Systems with Emergent Collective Computational Abilities*, Proc. Natl. Acad. Sci. U.S.A. **79**, 2554 (1982).

[24] E. Gardner, *The Space of Interactions in Neural Network Models*, J. Phys. A **21**, 257 (1988).

[25] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, *Overcoming Catastrophic Forgetting in Neural Networks*, Proc. Natl. Acad. Sci. U.S.A. **114**, 3521 (2017).

[26] J. W. Rocks, H. Ronellenfitsch, A. J. Liu, S. R. Nagel, and E. Katifori, *Limits of Multifunctionality in Tunable Networks*, Proc. Natl. Acad. Sci. U.S.A. **116**, 2506 (2019).

[27] D. Hexner, A. J. Liu, and S. R. Nagel, *Role of Local Response in Manipulating the Elastic Properties of Disordered Solids by Bond Removal*, Soft Matter **14**, 312 (2018).

[28] M. Isobe and K. Okumura, *Initial Rigid Response and Softening Transition of Highly Stretchable Kirigami Sheet Materials*, Sci. Rep. **6**, 24758 (2016).

[29] J. Ma, J. Song, and Y. Chen, *An Origami-Inspired Structure with Graded Stiffness*, Int. J. Mech. Sci. **136**, 134 (2018).

[30] M. Dogterom and T. Surrey, *Microtubule Organization In Vitro*, Curr. Opin. Cell Biol. **25**, 23 (2013).

[31] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation* (Addison-Wesley/Addison-Wesley Longman, Reading, MA, 1991).

[32] D. J. Amit, H. Gutfreund, and H. Sompolinsky, *Storing Infinite Numbers of Patterns in a Spin-Glass Model of Neural Networks*, Phys. Rev. Lett. **55**, 1530 (1985).
[33] D. J. Amit, H. Gutfreund, and H. Sompolinsky, *Spin-Glass Models of Neural Networks*, Phys. Rev. A **32**, 1007 (1985).
[34] R. Tibshirani, *Regression Shrinkage, and Selection via the Lasso*, J. R. Stat. Soc., Ser. B **58**, 267 (1996).
[35] H. Zou and T. Hastie, *Regularization and Variable Selection via the Elastic Net*, J. R. Stat. Soc., Ser. B **67**, 301 (2005).
[36] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical Learning with Sparsity: The Lasso and Generalizations* (CRC Press, Boca Raton, FL, 2015).
[37] Y. LeCun, C. Cortes, and C. J. Burges, *MNIST Handwritten Digit Database*, http://yann.lecun.com/exdb/mnist.
[38] N. Pashine, D. Hexner, A. J. Liu, and S. R. Nagel, *Directed Aging, Memory, and Natures Greed*, Sci. Adv. **5**, eaax4215 (2019).
[39] O. Chaudhuri, S. H. Parekh, and D. A. Fletcher, *Reversible Stress Softening of Actin Networks*, Nature (London) **445**, 295 (2007).
[40] F. J. Lockett, *Nonlinear Viscoelastic Solids* (Academic Press, London, 1972).
[41] J. D. Humphries, P. Wang, C. Streuli, B. Geiger, M. J. Humphries, and C. Ballestrem, *Vinculin Controls Focal Adhesion Formation by Direct Interactions with Talin and Actin*, J. Cell Biol. **179**, 1043 (2007).
[42] S. Li, P. He, J. Dong, Z. Guo, and L. Dai, *DNA-Directed Self-Assembling of Carbon Nanotubes*, J. Am. Chem. Soc. **127**, 14 (2005).
[43] J. D. Hartgerink, J. R. Granja, R. A. Milligan, and M. R. Ghadiri, *Self-Assembling Peptide Nanotubes*, J. Am. Chem. Soc. **118**, 43 (1996).
[44] W. J. Blau and A. J. Fleming, *Materials Science. Designer Nanotubes by Molecular Self-Assembly*, Science **304**, 1457 (2004).
[45] A. Javanmard and A. Montanari, *Localization from Incomplete Noisy Distance Measurements*, Found. Comput. Math. **13**, 297 (2013).
[46] M. Stern, C. Arinze, L. Perez, S. E. Palmer, and A. Murugan, *Supervised Learning through Physical Changes in a Mechanical System*, Proc. Natl. Acad. Sci. U.S.A. **117**, 14843 (2020).
[47] J. C. Maxwell, *L. On the Calculation of the Equilibrium and Stiffness of Frames*, London, Edinburgh, Dublin Philos. Mag. J. Sci. **27**, 294 (1864).
[48] J. D. Paulsen and N. C. Keim, *Minimal descriptions of cyclic memories*, Proc. R. Soc. A **475**, 20180874 (2019).