THE UNIVERSITY OF CHICAGO

DESIGN AND LEARNING IN MECHANICAL SYSTEMS

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF PHYSICS

BY
MENACHEM STERN

CHICAGO, ILLINOIS
DECEMBER 2019

In dedication to my beloved family, whose support made it all possible.

‘Tell me and I forget, teach me and I may remember, involve me and I learn.’

– Benjamin Franklin

Table of Contents

List of Figures

Acknowledgments

First of all I would like to thank my advisor, Prof. Arvind Murugan, for being a mentor and a friend, for his encouragement, guidance, and patience. Few are the people who could grant so much trust and inspiration in the study of such outlandish ideas. This work would not have been possible without his support, for which I am extremely grateful.

I would also like to thank Dr. Matthew Pinson, who was a close collaborator on many of the projects presented here. For many discussions, suggestions and encouraging words, I am indebted to him and other collaborators, in particular Chukwunonso Arinze and Viraaj Jayaram, who played important roles in the study of design and learning in origami.

I am grateful to Prof. Thomas Witten, who has been a major source of support for the entire time. He was always open to hearing me out, and kindly allowed me to regularly join and present in his group meeting. His openness and attentiveness made me feel welcome in the UChicago soft matter community, and I consider my discussions with him to have been especially enjoyable.

I am thankful to Prof. Wendy Zhang, my research advisor for the first year in Chicago, as well as Profs. Heinrich Jaeger and Ivo Peters (U. of Southampton), with whom I have worked closely on the hydrodynamics of suspension droplets. Their advice and support allowed me to experience and enjoy a much broader scope of soft matter physics research. I would also like to thank my thesis committee members, Profs. Dam Son and Savdeep Sethi for the support and discussions on different aspects of the work.

Finally, I would like to especially thank the dear members of my family. My mother, Hava, for her never-ending support, even from faraway. My wife, Shahar, for her love, support and

encouragement during the periods together and apart. For the great discussions on physics, biology and statistics, and feeling of home no matter where we went. Our daughter Talia who joined halfway through, for her cheerfulness, inspiration, and limitless energy. Their love and support were vital for the success of my studies.

Abstract

For millennia, people have designed diverse machines to perform countless different tasks. *Design* – the creation of a system according to a *rational plan*, is so ubiquitous in the engineering of mechanical systems, that the word became synonymous with the final engineered product. However, recent advances in neuroscience and computer science suggest a different approach to constructing mechanical systems, namely *learning*. If a system can modify the properties of its components in response to external inputs, it may be able to learn desired behaviors by observing examples of use. Learning mechanical systems may have distinct advantages over designed systems, such as the potential to be trained for a task by an end-user rather than a designer, and the ability to adapt to new tasks while still capable of accomplishing previously established ones.

In this work, we study and compare design and learning approaches in two types of mechanical systems, self-folding origami and elastic networks. By utilizing an energy-based viewpoint, we show how these systems are designed to perform certain tasks (e.g. folding in a desired way, or having predefined multi-stability), and how they can learn to perform such tasks by experiencing examples of use. We elucidate the distinct advantages and disadvantages of design and learning approaches in these specific systems. Finally, we lay out explicit analogies between learning mechanical systems and learning in neuroscience and computer science. Thus, we hope that future mechanical engineering disciplines will exploit the surge in learning theory to create new classes of learning machines, capable of feats yet impervious to traditional design frameworks.

Chapter 1

Introduction

Mechanical systems, or machines, convert energy to relative motion of their constituents, in order to perform a certain function. These systems are ubiquitous in nature and technology. Included in this category are the simplest machines employed by humans since antiquity, such as the axe, lever and wheel, up to intricate systems like evolved proteins and man-made power plants. The study, understanding, and ultimately engineering of these systems had contributed immeasurably to society. In recent years physical machines have been somewhat overshadowed by the information age computation based systems, that are electronic in nature and devoid of moving (macroscopic) parts. However, there is no doubt that mechanical systems will continue to have a crucial role in human experience.

Though mechanical systems architectures, as well as the tasks they perform, are extremely diverse, we may divide machines into two broad classes according to the principles by which they are created. The everyday objects we interact with and refer to as machines are usually *designed* specifically to perform a certain task [1]. One simple example is the collapsible umbrella; it opens up in a swift, predictable motion to protect the user from the rain, and when not in use may be collapsed back to allow for compact storage. How is such a machine designed? It is clear that a random collection of rods, hinges and pieces of fabric do not naturally form a working umbrella. Instead, these different components are brought together by rational *design principles*.

The designer moves the components around, connects them in different ways, and even replaces them altogether, in order to obtain a final product that achieves the desired functionality. Though often aided by trial and error and previous experience, the design process focuses on optimizing the mechanical system and its interacting components in order to facilitate the system's goals.

While the majority of engineered systems around us are designed, living systems (that can also be regarded as mechanical), are generically *evolved*. A typical example for such a system is an animal, evolved through mutation and natural selection to be fit (able to thrive and reproduce) in its natural environment [2]. In contrast to designed systems, an evolved system does not change its components via a rational optimization framework. Instead, animal individuals are born and either succeed or fail in rearing offspring according to their intrinsic fitness (and chance). In other words, the evolution process produces many individual examples of animals, and proceeds by proliferating the fittest examples. Following the scientific description of systems as evolving, analogies have been drawn between evolution and the process of *learning* [3]. While an animal learns, the structure of its brain changes in a way that allows it to acquire new memories or abilities. Crucially, these modifications are facilitated by sensory input, or examples (similarly to evolution); we cannot learn information we were never exposed to.

This idea of *learning*, or utilizing examples of a desired task in order to 'train' the system to perform that task, has been embraced by the fields of statistics, and more recently, computer science (machine learning) [4]. These fields make use of the vast quantities of data, available through modern monitoring and computation, to train algorithms for various tasks, from distinguishing handwritten digits to predicting the price of stocks. Though learning frameworks are now routine in data science, these ideas have only recently been explored in engineering [5], and have so far focused mostly on application of machine learning algorithms to analyze engineered systems.

We propose that the ideas of learning theory could be applied to mechanical systems in a novel fashion. "Smart materials" [6] that change in response to external manipulation may be used as learning models. In other words, these materials can be trained to perform desired tasks just by observing examples and modifying their structure accordingly. This approach of training

mechanical systems for specific tasks has only recently been considered [7, 8], and it may have considerable advantaged in comparison to the traditional design approach. In particular, while design requires expert knowledge of the system at hand in order to rationally modify it for a desired task, training by example may be done by an end-user with a specific desired task in mind.

In this work we study and compare design and learning approaches for mechanical systems. We address two distinct systems that have seen many applications in engineering and art: Self-folding origami and elastic networks. Origami is a mechanical network described by a crease pattern on a thin flat material sheet. Origami can be actuated by folding the creases, giving rise to intricate $3d$ structures, possibly fit for specific functions (e.g. paper plane). Self-folding origami is a special class of origami, in which the topology of the crease pattern constrains all of the creases to fold simultaneously. In principle, this means the entire pattern can be actuated by folding just one crease. This type of origami structures, discovered relatively recently [9], has garnered much interest in the engineering community for their possible advantages. These include easy manufacturing (etching creases on a thin $2d$ sheet) and deployment (few actuators necessary to fold the entire structure). Furthermore, for thin sheets, the folding of origami depends only on the geometry of the pattern (and not on the material) so that engineered application of self-folding origami were realized over many length scales [10]. Though studied for decades in engineering, mathematics and arts, the study of self-folding origami with physics based approaches is relatively new [11]. This line of research has focused on the characterization of folding in self-folding sheets [12, 13] and the design of particular target folded shapes [14] (Fig. 1.1(a)).

In contrast to origami, an elastic network is a general concept that refers to any system that could be modeled as a collection of nodes connected by springs. Most materials, natural or manufactured, indeed belong to this category. Compound mechanical structures, like the collapsible umbrella discussed above, can also be considered as elastic networks. Thus the relevance of this type of models to our everyday experience is hard to overstate. Though most elastic networks change predictably under small strains (in the linear response regime), they may have vastly different responses to large strains, depending on their microscopic structure and topology [16]. When

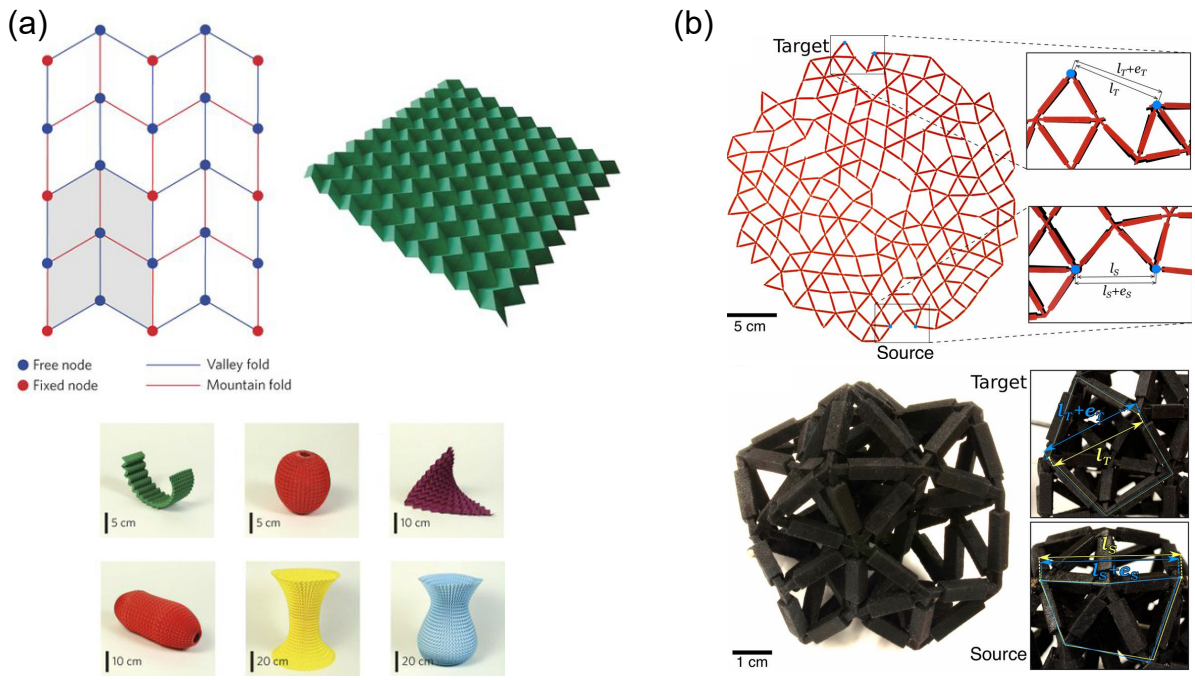

Figure 1.1: Self-folding origami and Elastic networks
(a) (Top) The famous Miura pattern where all creases fold simultaneously to a compact shape. (Bottom) Different folded designs obtainable by small variations of the Miura pattern (adapted from Figs. 1,2 of Ref. [14]). (b) Two and three dimensional elastic networks manufactured using plastic bonds and designed to exhibit long range (allosteric) actuation modes (adapted from Fig. 4 of Ref. [15]).

such structures are constructed of macroscopic parts, they are often renamed "metamaterials" [17], with impressive modularity that allows for the design of intricate desired responses. Indeed, elastic networks have recently been designed to exhibit remarkable properties such as auxetic [18] and allosteric (long range) modes [15], as seen in Fig. 1.1(b). It is important to note that many of these scientific achievements were possible by the study and imitation of elastic networks in living systems, such as proteins and cytoskeletons, that exhibit a variety of unique mechanical responses [19].

To study these origami and elastic systems, we adopt an energy based point of view. Our analyses are based on exploring the configuration space of the system (i.e. a space where each dimension corresponds to the available states of each degree of freedom), and assigning an energy value to every configuration. Pictorially, one may imagine a high-dimensional 'energy landscape' that may be very complex, with a large number of important features such as local minima, maxima and saddle points of varying order [20]. In equilibrium, the system will occupy one of the local minima in the energy landscape. A soft mode may be defined as a relatively low energy valley connected to a local minimum, such that actuating the system along this mode is easy in practice. With this energy landscape framework in mind, the process of creating a mechanical system for a specific goal becomes a question of sculpting such high dimensional energy landscapes, subject to physical limitations.

As discussed above, energy landscapes may be adapted by either rational considerations (design), or the system experiencing examples of use (training, or learning). This work will explore methods of design and learning to create mechanical systems with desired properties, or fit to perform certain tasks. The work will be divided into five main chapters, each of which is a reprint of one paper, either published or submitted for publication. The majority of chapters focus on design and learning in self-folding origami, while the last main chapter centers on elastic networks. In the following we detail the contents of these chapters. The final chapter will be reserved for a discussion.

Chapter 2 serves as a more thorough introduction to self-folding origami and its description

using energy based models. Most known origami patterns with a self-folding topology are either completely soft (having zero-energy folding modes) or completely rigid (not folding at all). We study the creation of disordered self-folding patterns with an arbitrary chosen folding energy scale, bridging the gap between the two extremes. This is done by using a design approach, i.e. changing the pattern geometry using a rational process (namely, optimizing the residues of some 'loop equations') in order to tune the folding energy. This chapter is a reprint of Ref. [21].

Chapter 3 discusses self-folding origami as an example of a complex (glassy) system. We show that the flat, unfolded state, connects exponentially many distinct folding modes, such that actuating the folding motion one is interested in effectively requires a great deal of control. From an energy perspective, the landscape of origami folding is glassy, with exponentially many local minima. By employing computational strategies of solving NP-complete problems (e.g. Sudoku), we suggest a 'folding island' algorithm that allows the correct folding of origami with just a few actuators. This chapter is a reprint of Ref. [22]. Though in itself not directly related to design or learning frameworks, this chapter introduces the idea of folding complexity that is crucial for the next two chapters.

Chapter 4 studies a design approach for rescuing 'self-foldability' in self-folding origami. We earlier established that folding origami is an exponentially hard (NP-complete) problem for sheets with free folding creases. We find that if creases can have a heterogeneous stiffness profile, one can modify (design) the folding pathway topology in interesting ways. For example, unwanted folding modes may be eliminated from the flat state fork by means of saddle-node bifurcations. We show that one can optimize the stiffness profile to prefer a single folding motion, effectively eliminating all others. Such sheets have their self-folding property rescued, as any forces applied to them will actuate only the desired folding motion. This chapter is a reprint of Ref. [23].

Chapter 5 proposes self-folding origami as a system that can be trained with an analogy of supervised learning. We show that self-folding sheets, folded using sets of 'training forces', can be programmed to exhibit a desired force-fold map. In analogy to machine learning algorithms that distinguish cats and dogs in images, our sheets effectively act as force classifiers, able to

learn distinctions between sets of applied forces. The analogy to learning algorithms goes deeper, as we show that these trained sheet classifiers are able to generalize (correctly classify unseen 'test' forces). The problem of achieving the desired force-fold map (classification problem) is approached using a learning framework, as a design approach to achieving such results is hard to define in general. This chapter is a reprint of Ref. [24].

Chapter 6 compares design and learning approaches for solving the same problem, namely, how to obtain an elastic network with multiple predefined stable states. By design, multi-stability may be obtained by optimizing the stiffness coefficients in a network of linear springs, using solutions to sets of linear equations. We find that learning these multiple stable states in a growing network requires the use of completely different, non-linear springs, to stabilize the desired states. This result is connected to the idea of sparse regression in statistics, where springs associated with each individual stable state only store information about that state. The advantage of learning multi-stability is in the flexibility of this method. The system can learn new states continually, where new stable states can be learned in the network while retaining old ones. In comparison, our design approach to multi-stability requires complete redesign of the network whenever a single new state is to be added. This chapter is a reprint of Ref. [25].

Chapter 2

Self-folding origami at any energy scale

Programmed instabilities and weak spots have emerged as a powerful tool to design a unique preferred deformation mode into mechanical structures [26, 27]. Such mechanisms are attractive in actuators [28, 29], meta-materials [30], art, architecture [31, 32], robotics [33] and other applications at different length scales because mechanisms require minimal control at the time of deployment; as seen in folding chairs or unfolding umbrellas, the designed deformation is a unique one-dimensional path in configuration space through which the structure is naturally guided under any external force. Mechanisms [34], even more so than marginal structures [35], are delicately poised at the boundary between being rigid and floppy. Despite much recent interest in large extended mechanisms [10, 14, 31, 36, 37, 38, 39] and some critical contributions towards the same [36, 40, 41, 42, 43, 44], most work has focused on deformations with high symmetry, and the space of designed disordered deformations remains largely unexplored.

A prominent and ancient example of designed deformation is origami. In particular, rigid origami is the study of stiff sheets that do not bend except at the prescribed creases [36]. If creases are placed at just the correct angles relative to each other, the sheet as a whole has exactly one allowed deformation in which all the creases fold at the same time. Such sheets can be described [45] as self-folding because the allowed mode will be actuated by almost any applied force; there is no need to precisely tailor the folding forces. While a general origami pattern might have several

folding motions, a self-folding pattern will have a unique extended motion that requires less energy than all others.

However, even in this well-studied area, most known examples of self-folding crease patterns are in fact rigidly-foldable (i.e., foldable at precisely zero energy cost). With the exception of some influential works discussed below [36, 40, 41, 42, 43, 44], rigidly-foldable crease patterns are often periodic structures made of repeating units, such as Miura Ori and its derivatives [12, 13]. Further, origami design has often been limited to the mountain-valley (MV) pattern implicit in Miura-Ori [9, 14, 36]. Many such studies of rigid origami have also been restricted to so-called flat-foldable or near-flat-foldable vertices [36, 46] (i.e. patterns in which all creases fold to angle π simultaneously); the flat-foldability restriction on angles in a crease pattern leads to dramatic algebraic simplifications in rigidity calculations. As a result, Miura-Ori derivatives are often rigidly foldable, with the stiff sheet between creases (i.e., the 'faces') not bending at all when the creases are folded.

Restricting study to the rigid foldable patterns with no face bending misses a larger space of near-perfect mechanisms in which face bending or energy cost of actuation can be made arbitrarily small. Understanding the full space of crease patterns as a function of folding energy is also crucial for self-folding origami applications [10], since applications vary widely in material stiffness and actuation torques (or energies) available. For example, folding a structure made of stiff plates connected by Shape-Memory Polymer hinges [47, 48] that provide low actuation torques might require nearly-rigid foldable patterns; but using Shape-Memory Alloys [49] or ionic electroactive polymer [10] hinges that provide higher torques would allow use of less foldable patterns as well. Similarly, one might wish to prevent accidental deployment of a self-folding hydrogel capsule [50] due to small pH fluctuations, necessitating less foldable patterns.

Surprisingly little is known about general self-folding origami patterns that are not exactly rigidly foldable. Important contributions include Huffman's work [41] on general n-valent vertices and Tachi's simulation scheme of origami patterns [36]. Wu et. al. introduced analytic methods to analyze motions of multi-vertex patterns [44], extending Belcastro and Hull's condition for testing

rigid-foldability [42] for non-flat foldable patterns. Tachi went beyond rigid foldability for general patterns by establishing design principles for first order foldability [40, 43].

Energy scale-dependent origami design and statistical properties of typical patterns are the basic building blocks needed for a physically motivated theory of origami [39], relevant to both natural [29] and synthetic [10, 51] systems. In this work, we present a systematic exploration of the space of self-folding crease patterns as a function of folding energy by solving equations in sequence. We further show that Mountain-Valley choices strongly affect foldability; e.g., 62% of all Mountain-Valley choices account for 10% of highly foldable patterns. Finally, we find an entropy-energy relationship quantifying the number of crease patterns with given folding energy, describing how many more crease patterns become available for a given increase in available actuation energy, e.g., in active hinges [10].

2.1 Results

2.1.1 Vertex transfer function

As in past work [12, 14, 36], we study patterns made of general 4-vertices, like those shown in Fig. 2.1, since vertices with three or fewer edges are completely rigid while vertices with more than four edges are too soft (i.e., have multiple continuous degrees of freedom). Assuming the angles $\theta_{12}, \theta_{23}, \theta_{34}, \theta_{41}$ between creases of the vertex add to 2π, we note two primary facts about generalized 4-vertices studied earlier [12]; Three of the four creases must fold in a common orientation (say, Mountain, black in Fig. 2.2a) with the final odd-one-out crease folding the other way (Valley state, red). The final odd-one-out crease can be any one of the two creases whose neighboring angles add to less than π [12] (Fig. 2.6). Once the discrete odd-one-out choice in Mountain-Valley has been made, a 4-vertex has exactly one folding degree of freedom (Fig. 2.2a); the folding angle ρ_i at any crease i completely determines any other folding angle ρ_j. For two

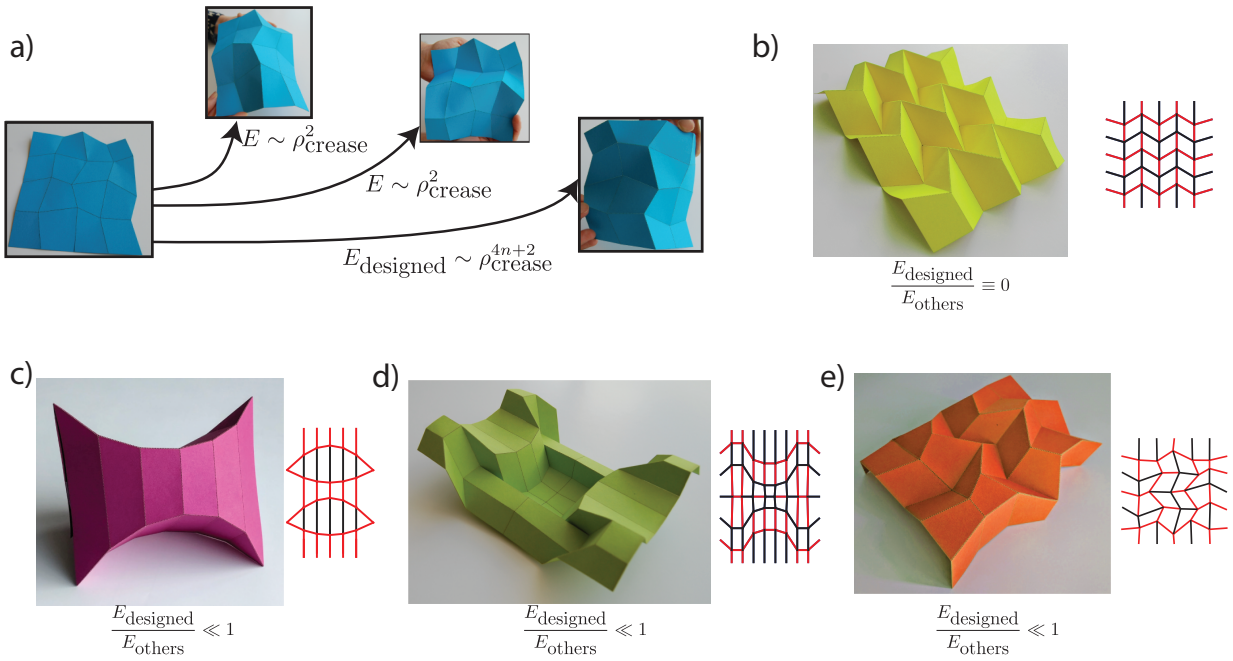

Figure 2.1: Designing self-folding origami

(a) Forces applied to a 'self-folding' sheet will preferentially actuate the one pathway designed to have significantly less face bending than the other two pathways shown (i.e., designed to have $E_{\text{designed}}/E_{\text{others}} \ll 1$). (b) The celebrated Miura Ori pattern is a special highly symmetric pattern with $E_{\text{designed}}/E_{\text{others}} \equiv 0$. (c,d,e) We study a larger space of experimentally relevant crease patterns by going beyond rigidly foldable symmetric patterns. The folding energy scale of such patterns can be made as small as needed in a systematic manner; $E_{\text{designed}} \sim \rho^{4n+2}_{\text{crease}}$ where ρ_{crease} is the median crease folding angle, and n the number of solved loop equations that are derived here. Patterns in (c,d,e) are geometrically distinct from traditionally studied limits (Kawasaki vertices, Miura-Ori Mountain Valley choice). These patterns solve exactly only one (d,e) or two (c) loop equations.

chosen adjacent creases, symbolically write,

$$\rho_1 = T(\rho_2; \{\theta\}) \tag{2.1}$$

where $\{\theta\}$ are the four in-plane angles between creases.

For small fold angles ρ_i, we can linearize the above relationship and write

$$\rho_1 \approx R(\{\theta\})\rho_2 + O(\rho_2^2). \tag{2.2}$$

R's determine the mechanical advantage and dynamic range of folding angles at a vertex.

Similar transfer functions have appeared in the literature over the years [34, 36, 39, 41, 52, 53]. We emphasize that the transfer functions T, R depend on the Mountain-Valley configuration at the vertex [12]. Explicit forms of T, R for general 4-vertices, including their MV dependence, are presented in Supplementary Note 1.

2.1.2 Loop equation and tuneable stiffness

While a single 4-vertex (Fig. 2.2a) always has one degree of freedom, four vertices linked to form a quad are generically rigid. In fact, the number of folding degrees of freedom (i.e., 12 folding angles ρ_i) exactly matches the number of constraints relating these folding angles (3 at each vertex). Hence a generic quad has, at best, a discrete set of folded states — the folding motion between such states will generically involve face bending or other such violation of constraints.

Thus, smooth folding motions (modes) require fine-tuning of the in-plane angles at each vertex ('design parameters'). An intuitive way to understand the fine tuning required is to write a consistency loop condition for a fold angle ρ, say that of AD (see Fig. 2.2b), transported around the quad (i.e. forming a closed loop),

$$\rho = T^{\mathrm{D}}(T^{\mathrm{C}}(T^{\mathrm{B}}(T^{\mathrm{A}}(\rho)))) \tag{2.3}$$

This nonlinear loop equation needs to be satisfied as a function of ρ, not just at particular values of

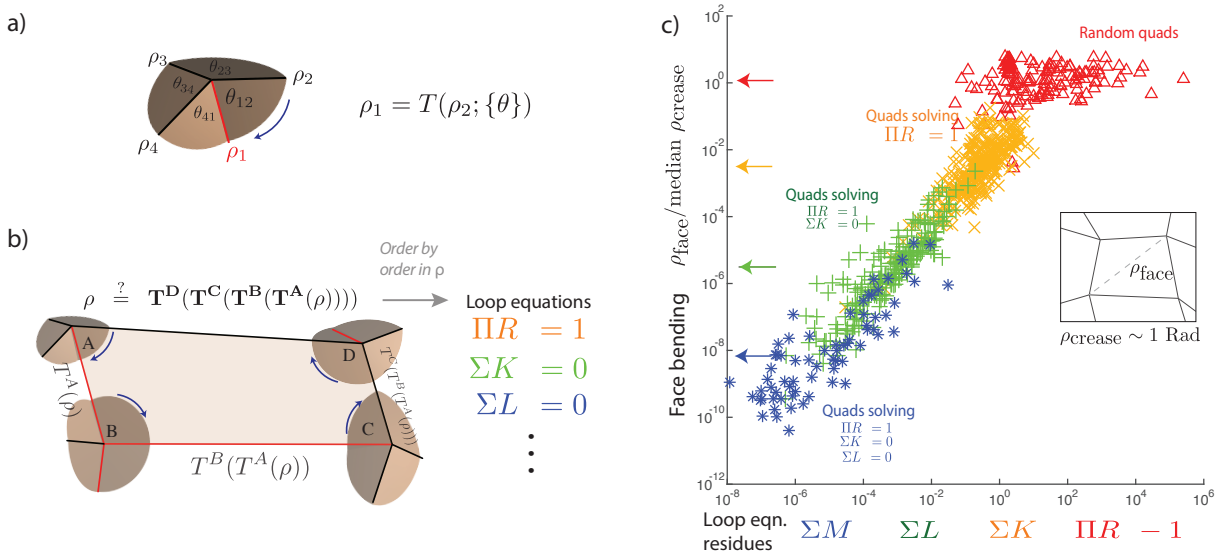

Figure 2.2: Loop equations uncover folding modes of variable face bending over orders of magnitude.
(a) Folding angles ρ_1, ρ_2 of adjacent creases at a 4-vertex are related by a transfer function T, determined by in-plane angles θ. (b) For a quad to be foldable, a fold angle ρ must return as ρ when transported around the loop using transfer functions $T^{\mathrm{A}}, T^{\mathrm{B}}, T^{\mathrm{C}}, T^{\mathrm{D}}$. For a smooth folding motion, the equation must be satisfied order by order in ρ. (c) Face bending can be dramatically reduced in a controlled manner by solving loop equations in sequence. Random quads (red triangles), not designed to solve any loop equation, show face bending comparable to crease folding. Quads solving the first loop equation $\Pi R = 1$ (orange $\times$s) typically have face bending $< 10^{-2}$ Rad. The residue of the highest loop equation *not* solved determines the extent of face bending; hence orange points and green points show the drop of face bending with decreasing ΣK and ΣL respectively.

ρ, in order to have a smooth folding mode. Taylor expanding the right hand side and subtracting ρ,

$$0 \equiv f_1\rho + f_2\rho^2 + f_3\rho^3 + \dots \tag{2.4}$$

where $f_1 = \Pi R - 1$. Setting $f_i = 0 \ \forall i$ gives - potentially - an infinite set of equations for the design parameters (i.e., in-plane angles θs) - to have a folding motion to all orders in ρ. Similar loop equations for lowest order foldability were derived by Tachi [40, 43] earlier.

We can write the series of loop equations, defined term by term using the expansion of the transfer function of Equation (2.4). The loop equations are computed explicitly in Supplementary Note 1, while here we write them symbolically as:

$$\begin{aligned} \Pi R: &\qquad R^{\mathrm{A}}R^{\mathrm{B}}R^{\mathrm{C}}R^{\mathrm{D}} = 1 \\ \Sigma K: &\qquad K^{\mathrm{A}} + K^{\mathrm{B}} + K^{\mathrm{C}} + K^{\mathrm{D}} = 0 \\ \Sigma L: &\qquad L^{\mathrm{A}} + L^{\mathrm{B}} + L^{\mathrm{C}} + L^{\mathrm{D}} = 0. \\ &\vdots \end{aligned} \tag{2.5}$$

As shown in Supplementary Note 1, R^{V} is a property of in-plane angles at a single vertex V. $K^{\mathrm{V}}, L^{\mathrm{V}}, \dots$ are products of functions of a single vertex V and of $R_{\mathrm{V}'}$ at other vertices $\mathrm{V} \neq \mathrm{V}'$.

MATLAB Code to compute loop equations to arbitrary order is given as Supplementary Material. For a quad, we verified that the first five equations are independent. Combined with Tachi's earlier work [36] that discovered a 6-parameter family of rigidly foldable quads with a special symmetry (flat foldability), our work suggests that only the first 5 loop equations are fully independent (See Supplementary Note 2), as each loop equation constrains one parameter of the 11d $\{\theta\}$ design space. Here we focus on exploring the full space of creases patterns as a function of foldability and Mountain-Valley choices.

When a quad does not satisfy all loop equations exactly, there is no perfect zero-energy mode. Allowing a single diagonal fold (Fig. 2.2c inset) adds an additional degree of freedom and thus

allows any augmented quad (a quad with an additional face diagonal crease) to fold. Measuring the angle ρ_{face} of a freely folding diagonal is a proxy for the face bending energy in the presence of a stiff face (Fig. 2.7).

We note that stretching energy in thin sheets scales the same way with ρ_{face} as bending energy due to a virial theorem [54, 55], but is expected to be considerably smaller [54, 55]. Further, for thin sheets, bending strain is much larger than stretching strain in low energy configurations [54, 55]. Hence, in the following, we model both the energy and geometry by considering only face bending. We later check the validity of this thin sheet approximation using finite element simulations in COMSOL (see Fig. 2.5 and Fig. 2.7). Thickness in real application varies, e.g., 0.05 mm thick NiTi sheets of width 50mm for stents [49] to 1 um thick GaAs sheets of width 100 um [56] for optically actuated mirrors.

To study the relationship between loop equations and face bending quantitatively, we generated random quadrilateral patterns with random Mountain-Valley assignments and used them to solve loop equations order by order using gradient descent. We then added a crease along the face diagonal (Fig. 2.2c inset), simulated folding of each augmented quad from the unfolded state through small folding angles. In this way, we find,

$$\rho_{\text{face}} = a_1|\Pi R - 1|\rho_{\text{crease}} + a_3|\Sigma K|\rho_{\text{crease}}^3 + a_5|\Sigma L|\rho_{\text{crease}}^5 + \dots \tag{2.6}$$

where $\rho_{\text{crease}} > 0$ is the median crease folding angle and the coefficient a_i depend on the details (i.e., θ_{ij}) of the quad. Thus, as noted in Fig. 2.1a, the energy required to actuate the designed mode,

$$E_{\text{designed}} \sim \rho_{\text{face}}^2 \sim \rho_{\text{crease}}^{4n+2},$$

drops rapidly with the number n of the exactly satisfied loop equations in the hierarchy. Applying the loop equation hierarchy to different seeds of pattern designs allows discovery of soft foldable patterns devoid of symmetries or order in space (Fig. 2.1c,d,e). Such soft patterns may have in-

teresting mechanical properties that distinguish them significantly from the well-studied Miura ori pattern (Fig. 2.1b).

Remarkably, the relationship between face bending ρ_{face} and n strongly persists even if face bending is determined after folding to a large angle $\rho_{\text{crease}} \sim 1$ Rad. As shown in Fig. 2.2c, the loop equations when solved in sequence provide a controlled and systematic reduction in face bending over 9 orders of magnitude. Solving each successive equation reduces face bending by a factor of $\sim 10^2$. In addition, the residue of the leading loop equation *not* exactly solved is highly predictive of face bending. Thus the value of ΣK is predictive of face bending for quads that solve $\Pi R = 1$ (orange $\times$) while ΣL is predictive of face bending for quads that solve $\Pi R = 1$ and $\Sigma K = 0$ (green +) and so on.

Equation (2.5) thus provides a simple design principle for exploring the crease patterns at any chosen folding energy scale over many orders of magnitude; one simply solves the hierarchy of loop equations to the extent needed. Note that if the creases themselves have non-zero folding energy (e.g., due to finite thickness), the folding energy E_{designed} would be bounded from below by such an energy scale; crease patterns cannot be made softer than the intrinsic stiffness of individual creases.

2.1.3 Mountain-Valley choice strongly affects foldability

The loop equations explicitly depend on the Mountain-Valley (MV) choices around the quad. The equations can be defined for any given MV choice, opening up the full space of origami patterns. Almost all work-to-date on origami is based on Miura-Ori's Mountain-Valley choice. In the following, we show that different MV choices lead to different typical foldability in a statistical sense.

We find that some MV choices are intrinsically more conducive to solving the loop equations than others. Hence we can categorize MV choices by foldability classes. To define these classes precisely, note that at each vertex, one can define the 'broken' direction to be the two longitudinal creases whose MV states differ (see key in Fig. 2.3a). The two creases in the orthogonal 'unbroken' direction have the same MV state. The crucial observation is that the creases in the unbroken

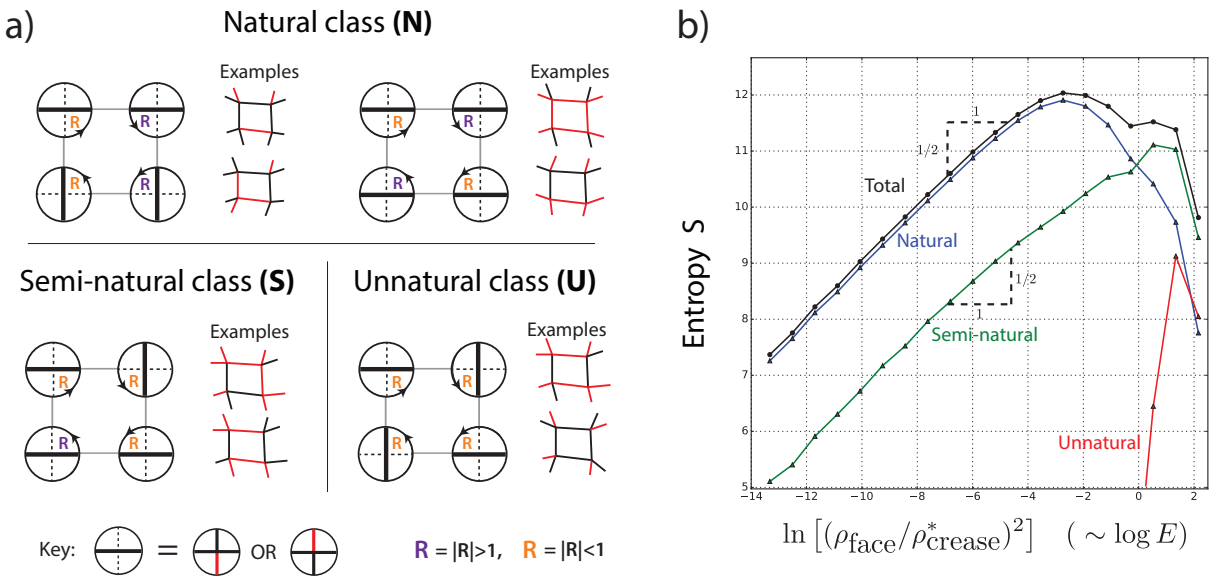

Figure 2.3: Mountain-Valley choices fall into three classes based on foldability of typical modes (a) For example, Semi-natural MV configurations dictate that $|R| > 1$ at three vertices and $|R| < 1$ at one, which is statistically less compatible with $\Pi R = 1$ than Natural configurations (two $|R| > 1$, two $|R| < 1$). (b) Consequently, when random modes of N, S and U types are sampled, U (Unnatural) type modes tend to be much stiffer than S or N type. We sampled 10^6 random modes, simulated folding and recorded their stiffness; we show the histogram binned by (log) face bending energy $\log \rho^2_{\text{face}}/\rho^{*2}_{\text{crease}} \sim \log E$, which we call the entropy $S(E)$. (ρ^*_{crease} = median ρ_{crease}). Among soft patterns ($E < 10^{-1}$), 90% of random modes are of Natural MV type which account for only 6/16th of all MV configurations. S and U type modes dominate at high energies $E > 10^{-1}$. The histogram captures a statistical relationship between MV choices and foldability for 'typical quad patterns and MV classes.

direction *typically* fold more than the broken creases; hence $|R_{ij}| \approx |\rho_i/\rho_j| < 1$ if i is in the broken direction and j unbroken (see Equation (2.2)).

Intuitively, some MV choices tend to make $|R| > 1$ at two of the four vertices around a quad and $|R| < 1$ at the other two. These Natural MV assignments (Fig. 2.3a) are most easily compatible with $\Pi R = 1$ (equation (2.5)). Semi-Natural MV patterns have $|R| > 1$ at three vertices and $|R| < 1$ at the fourth (or vice-versa). Finally, Unnatural quads have all four $|R| > 1$ (or $|R| < 1$); the in-plane θ angles of such a quad must be fine-tuned to be foldable.

The class of a MV choice thus determines how easy it is to solve the first loop equation $\Pi R = 1$. Random quads with Unnatural MV choice are far less foldable than Natural or Semi-natural types (see also Fig.2.8).

To quantify this statement, we sampled $\sim 10^6$ random quads by displacing the vertices of a regular square lattice randomly and independently. We then simulated folding each of these quads with a random folding torque to obtain a folding mode with $\max \rho_{\text{crease}} \sim 1$ Rad; we noted the resulting Mountain-Valley data as well as the resultant face bending ρ_{face} for each mode. The histogram, binned by face bending energy $\log E = \log \rho_{\text{face}}^2 + \text{const}$ is shown in Fig. 2.3b; we define the entries of this histogram to be the entropy $S(E)$ since $\int_{E_1}^{E_2} e^{S(E)} d \log E$ gives the number of modes in our random ensemble in the energy range $E_1 - E_2$ ($S(E)$ is defined only up to an additive normalization constant).

First, in Fig. 2.3b, we note that the total entropy follows a simple law up to quite stiff modes, $S(E) = \frac{1}{2} \log E$ that we explore further below. We also see that 90% of modes softer than $E \sim 10^{-1}$ are accounted for by Natural MV configurations, even though such configurations only account for 6/16 of all Mountain-Valley choices. Most of the remaining 10% of soft mdoes are accounted for by Semi-natural configurations (8/16 of all choices). Among stiff modes $E > 10^{-1}$, the situation is reversed and Semi-natural and Unnatural configurations form a majority.

Thus, in addition to opening the door to arbitrary MV choices, our work suggests previously unnoticed MV classes that qualitatively differ in their typical foldability. Such widely varying entropy of MV classes suggests important lessons in design; Natural configurations can be expected

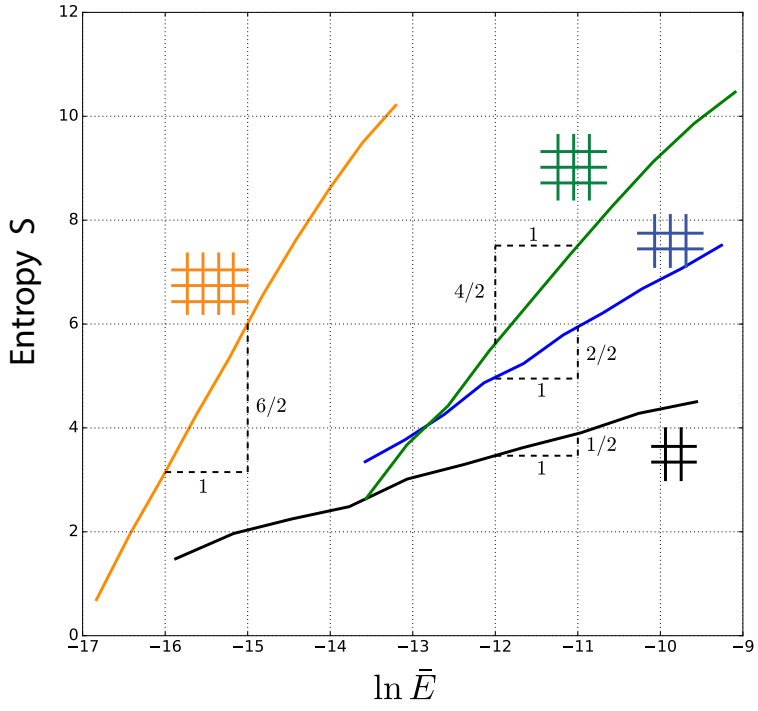

Figure 2.4: Entropy of crease patterns as a function of face bending energy
We sampled $\sim 10^5$ random quad meshes made of A quads (for $A = 1, 2, 4, 6$), folded them with the same fixed Mountain-Valley choice and noted the face bending energy per quad $\bar{E} \equiv (1/A)\sum \rho^2_{\text{face}}/\text{median}\,\rho^2_{\text{crease}}$. The entropy of patterns is $S = \frac{A}{2}\log \bar{E}$. ($e^{S(\bar{E})} d\log \bar{E}$ is the number of random patterns in an energy interval $d\log \bar{E}$.). Thus the probability of finding a soft crease pattern $e^{S(\bar{E})} d\log \bar{E} \sim \bar{E}^{\frac{A}{2}}$ in a random ensemble diminishes exponentially with mesh size A (for fixed $\bar{E}$) but only as a power law in energy $\bar{E}$ (for fixed A).

to be more forgiving of error in laying out creases while Unnatural configurations need to be highly fine-tuned to be foldable.

Entropy-energy relationship for large crease patterns

We have seen that a quad's folding mode can be made arbitrarily soft by solving a series of loop equations; but soft modes are rarer than stiff modes. In the following section we generalize these considerations to large origami meshes.

We sampled crease patterns made of A quads by displacing the vertices of a regular lattice with A inner faces randomly and independently, in much the same way as for the quad above. We

folded the resulting crease pattern using a torque that selects chosen MV data until $\max(\rho_{\text{crease}}) = 1$ *Rad* and recorded the resulting face bending ρ_{face} on each quad. We then made a histogram of $(1/A)\sum \rho^2_{\text{face}}/\text{median}\,\rho^2_{\text{crease}} \sim E/A \equiv \bar{E}$ from a large sampling of such lattices of different sizes; see Fig. 2.4.

We again define the entropy of crease patterns (now for a given MV) as the logarithm of the above histogram. We find that this entropy is extensive in pattern size A and has a simple form,

$$S(\bar{E}) = \frac{A}{2}\left(\log \frac{\bar{E}}{E_0}\right) + \dots \tag{2.7}$$

where $\bar{E}$ is the (intensive) face bending energy per quad, the ellipsis represent sub-leading corrections in A and E_0 is a constant discussed later. By construction, $e^{S(\bar{E})} d\log \bar{E}$ is the number of crease patterns of chosen MV with folding energy within an interval $d\log \bar{E}$ around $\bar{E}$.

We can understand the extensive scaling of entropy S with A and the $\log E$ dependence using the loop equations. As seen in Fig. 2c, for a single quad, face bending energy is simply related to loop equation residues, e.g., $\bar{E} \sim \rho^2_{face} \sim (\Sigma K)^2$ for the green points. On the other hand, we find that the fraction of quads in our random ensemble with loop residue less than $|\Sigma K|$ is simply proportional to $|\Sigma K|$; this is because patterns in our random ensemble appear to be uniformly distributed in their residues. Hence the total number of patterns of energy less than E scales as $\sqrt{E}$. Setting $\int_0^{\bar{E}} e^{S(\tilde{E})} d\log \tilde{E} \sim \sqrt{\bar{E}}$ (by our definition of entropy), we find $S(\bar{E}) = \frac{1}{2}\log \bar{E}$. (Note that the energy is not quite linear in the residue of the first loop equation $\Pi R - 1$ in Fig. 2c which is reflected in Fig. 2.3b as well.)

For large quad meshes, the above arguments apply to each quad since we need to solve loop equations independently for each quad in order to make softer patterns. For example, imposing a loop equation now removes A times as many design variables. Hence we find $S(\bar{E}) = (A/2)\log \bar{E}$.

Finally, note the $\log E$ dependence for large lattices in Equation (2.7) is expected to break down near an energy scale E_0 (the 10^5 samples generated here were not sufficient to probe this scale). In particular, Tachi's results [36] on rigid foldable patterns imply an entropy of $\sqrt{A}$ at zero energy.

Our investigations of entropy of folding modes as a function of energy connects to earlier work on crumpling transitions [57]; consider a sheet with a thermally variable crease pattern held at fixed temperature. At high temperatures, if the entropy of stiff modes is sufficiently high, the sheet might crumple for entropic reasons, even if energetically disfavored. Earlier analytic approaches were restricted to the entropy of rigid-foldable modes on regular lattices [58, 59] while our work is off-lattice and has a continuum energy E. Our entropy $S(E)$, based on quad meshes, only grows logarithmically in energy and hence does not show a first order transition.

The entropy-energy relationship in Equation (2.7) has theoretical and practical implications. In particular, the probability of a random pattern (when folded with a fixed MV) being softer than energy E decreases exponentially with mesh size A but only as a power law with E.

Such results are useful in understanding the trade-off between energy scales and design freedom. Self-folding origami applications vary greatly in the energy E_{material} needed to bend an uncreased face to a given angle; e.g., compare a Young's modulus of $\sim 10^3$ Pa for hydrogels [50] to $\sim 10^7$ Pa for NiTi alloy in origami stents [49]. Similarly, actuation mechanisms for active hinges are diverse, including electric [10], optical [56] , thermal [49] and chemical (pH) [50] methods. Hence, the actuation energy $E_{\text{actuation}}$ provided by active hinges (defined as work done by hinges during folding to 1 Rad) can vary widely; e.g., compare torques of $\sim 6 \cdot 10^{-3} Nm$ in 30 mm-long shape-memory polymer hinges [47, 48] to 5× or 400× that torque in ionic electroative polymers or shape-memory alloys respectively [10].

Taken together, $E_{\text{actuation}}/E_{\text{material}}$ can vary greatly across applications. Our energy-entropy relation shows that the fraction of all patterns suitable for such an application is
$\sim (E_{\text{actuation}}/E_{\text{material}})^{A/2}$ (for large A).

Additionally, micron-scale applications might often have a design requirement to prevent inadvertent actuation due to uncontrolled noisy processes of a lower energy scale E_{noise}; e.g., spontaneous temperature [49] or pH fluctuations in hydrogels [50] or random mechanical kicks. To avoid inadvertent actuation, the folding energy of patterns must be in the 'Goldilocks' zone between E_{noise} and $E_{\text{actuation}}$. The fraction of all patterns in the 'Goldilocks' zone can be computed to be

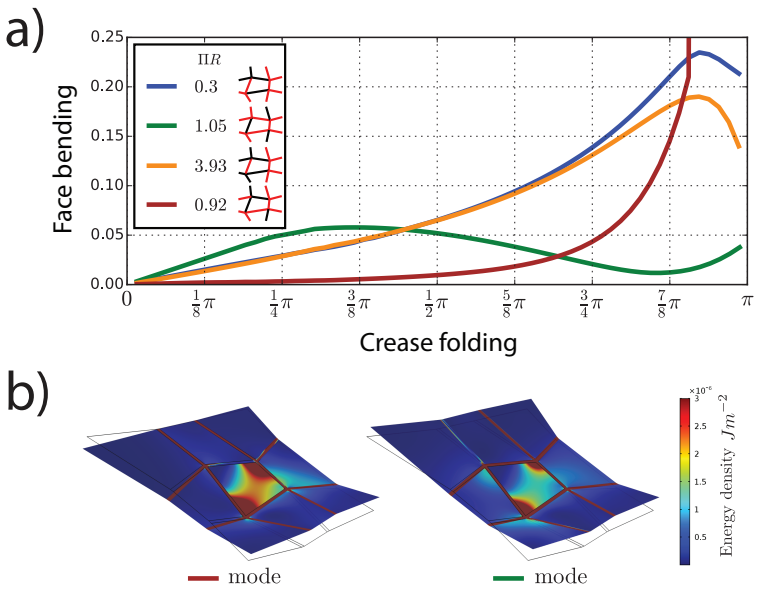

Figure 2.5: Face bending for large folding angles and in finite element simulations
(a) Face bending for a quad when folded along different Mountain-Valley modes. While stiffness for small folding angles is predicted by the loop equation residue $|\Pi R - 1|$, the initially-soft red mode becomes stiffer than others at large folding angles. The other three modes show non-monotonic bistable face bending. (b) Finite element simulations (COMSOL Multiphysics) of a 2D plate model of the same quad when folded along the red and green modes with a fixed magnitude of folding force. The associated face deformation elastic energy is localized to a diagonal furrow as predicted for thin plates [54]. (Material model: Melamine resin, Elastic modulus 6 GPa, density 1800 $kg\ m^{-3}$, thickness of plates is $\sim 1/1000$ of width.)

$(E_{\text{actuation}}/E_{\text{material}})^{A/2} - (E_{\text{noise}}/E_{\text{material}})^{A/2}$ for large A.

Equation (2.7) thus provides a basic guideline for how many more patterns become available if the actuation energy $E_{\text{actuation}}$ is raised, say, at the cost of higher power input[47] or if the energy of uncontrolled processes E_{noise} is lowered.

While our results were derived for the simplest random ensemble, they can be adapted to other ensembles of patterns relevant to specific applications.

Face bending along folding modes

In this work, face bending was measured by augmenting the quad with a diagonal crease (see Supplementary Methods) and then setting crease folding to one representative angle ($\sim$ 1 Rad) in

Equation (2.6). It is reasonable to expect face bending behaves as non-linear *function* of crease folding for large folding angles. In this section we study the face bending of folding modes for variable crease folding amplitudes.

In Fig. 2.5a, we show face bending as a function of crease folding for four different folding modes. We see that the loop equation residue $|\Pi R - 1|$ is a good predictor of face bending up to crease folding of $\sim \pi/2$ Rad. For larger folding, strongly non-linear effects kick in; the initially-soft red mode become stiffer than others rapidly while other modes show non-monotonic behavior.Non-monotonic bistable behavior has been seen before in experiments on highly symmetric flat-foldable patterns [13, 14].

To visualize what face bending stresses might look like in a real material without a diagonal crease, we show results of a finite element simulation in COMSOL in Fig. 2.5b of two select modes from Fig. 2.5a. Unlike our simplified model, this simulation accounts for stretching, bending of all 9 faces, finite thickness of the material and finite width of creases. While our simple diagonal crease model cannot capture the precise folding energies seen in the COMSOL simulation, we see that the bending stress is localized to a furrow along the diagonal, a result expected for thin sheets [54] (see Fig. 2.7 for more analysis and simulations). Further, when folded in COMSOL with a small but fixed folding force (note: not to fixed angle), the red mode shows higher stresses, implying that it is softer than the green mode, in agreement with our face bending model in Fig. 2.5a.

2.2 Discussion

In this work we have studied self-folding origami meshes as a function of folding energy, free from assumptions about Mountain-Valley data or symmetries such as flat-foldability. We found a design principle for self-folding patterns of arbitrary stiffness in terms of a series of loop equations applied to each quad in the pattern. These general patterns can exhibit diverse curvatures in 3 dimensions as compared to Miura-Ori. Related recent work [14] has achieved remarkable 3-dim structures by

gradual modulation of Miura-Ori patterns on length scales much larger than the repeating unit cell; however our work allows design on arbitrary length scales without being limited to any underlying repetitive motif.

Mountain-Valley (MV) data was found to greatly affect foldability. Natural MV types are typically much softer than Semi-natural or Unnatural types. This notion informs design decisions for soft self-folding modes, both as soft Natural modes are more numerous, but also as they are projected to be less prone to large stiffness fluctuations due to manufacturing errors.

We complemented these design principles with a statistical understanding of the space of all large quad meshes; we determined the total entropy of crease patterns of any given folding energy. Such a relationship tells us the number of patterns with a 'Goldilocks' folding energy that is lower than available actuation energy but high enough to prevent inadvertent actuation due to noisy uncontrolled processes.

While our work focused on quad meshes for concreteness, the loop equation hierarchy apply to any pattern made of arbitrary combinations of polygons, provided all vertices have valence four. We leave investigations of mechanisms with other topologies to future work.

In conclusion, understanding the space of crease patterns as a function of an energy scale combined with statistical results on the foldability of 'typical' patterns are crucial ingredients in developing a physically relevant theory of self-folding origami.

2.3 Supplementary Figures

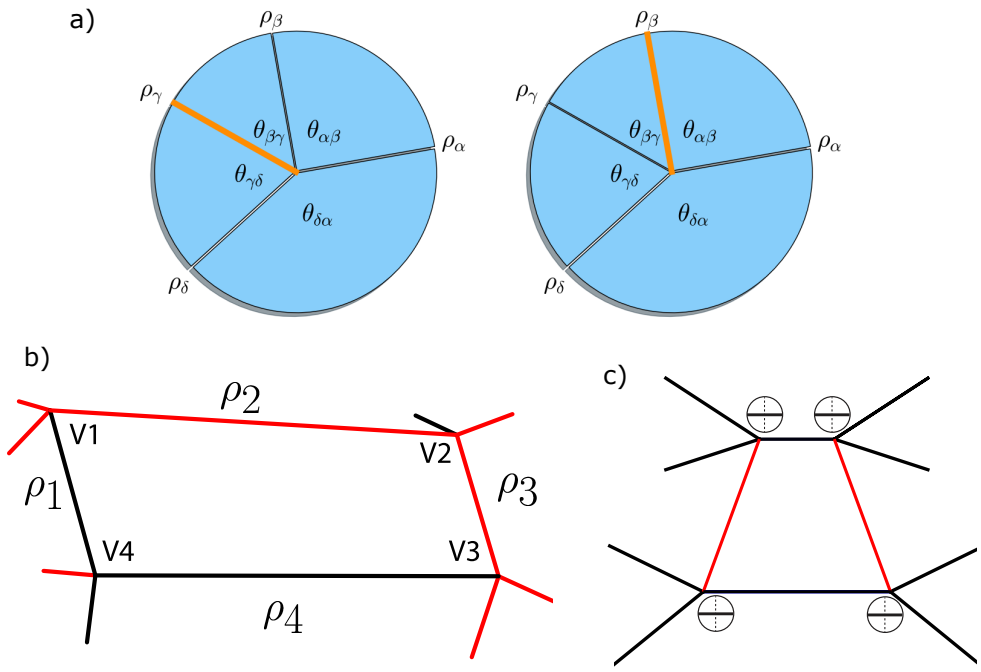

Figure 2.6: Modes of 4-vertices and quads
(a) A 4-Vertex is the intersection of four creases labeled $\alpha, \beta, \gamma, \delta$. The in-plane angles between creases labeled with θ. The 4-vertex has to zero-energy folding modes, both of which have an 'odd-one-out' crease whose MV state is opposite to the rest. The odd-one-out crease must be picked such that the two neighboring in-plane angles sum to less than π (orange creases). (b) A quad is compose of four 4-vertices connected in a loop. Putative folding modes are uniquely defined by noting the MV choices for each crease (black - mountain, red - valley). (c) A trapezoidal quad that is rigid-foldable due to symmetry.

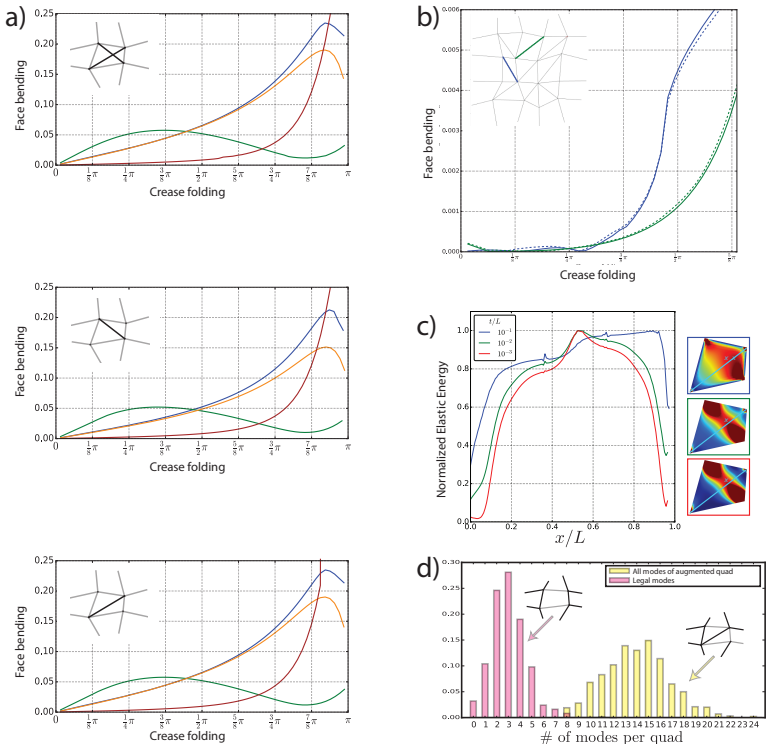

Figure 2.7: Folding of an added face diagonal is a robust proxy for face bending
(a) We augment the quad studied in Fig. 5 of the paper with different face diagonals, seen in each inset. Whether augmented with the usual face diagonal, the other face diagonal or an X-shape double diagonal, face folding along different modes reported in Fig. 5a of the paper remains qualitatively the same (For the X-shaped double diagonal, we plot the average of two face folding angles). Any augmentation that adds one degree of freedom to the simple quad does not change its folding modes. (b) Face diagonals (and thus loop equations) applied to individual quads are useful in designing large origami patterns. Face folding of quads that are part of a large pattern (solid lines) closely approximates the behavior of each quad when 'cut out' and folded in isolation (dashed lines). Loop equations, applied quad by quad to tune foldability, can be used to design large patterns of desired foldability. (c) Finite element simulations of the center plate show that face diagonals are good approximations for thin origami patterns. Applying boundary conditions of a folding mode, the elastic energy is increasingly localized to a diagonal furrow as the thickness t of the plate is decreased relative to lateral dimensions L (Young's modulus $Y = 3\ GPa$ (material = PVC), $L \sim 10cm$. Simulations using COMSOL). (d) Augmented quads have multiple folding modes with distinct Mountain-Valley types. We sampled 1000 augmented quads and made a histogram of total number of folding motions (yellow bars); typical augmented quads have ~14 distinct modes. However, many of these, when restricted to the simple quad, are not legal folding motions (e.g., in right inset, one vertex has 2 valley and 2 mountain folds). We only consider modes of the augmented quad that induce legal MV patterns on the simple quad; the histogram (pink bars) indicates ~3 such modes for typical augmented quads. Diagonal face creases have been used before to study deformations of Miura lattices [14] and bistability of flat foldable crease patterns [13].

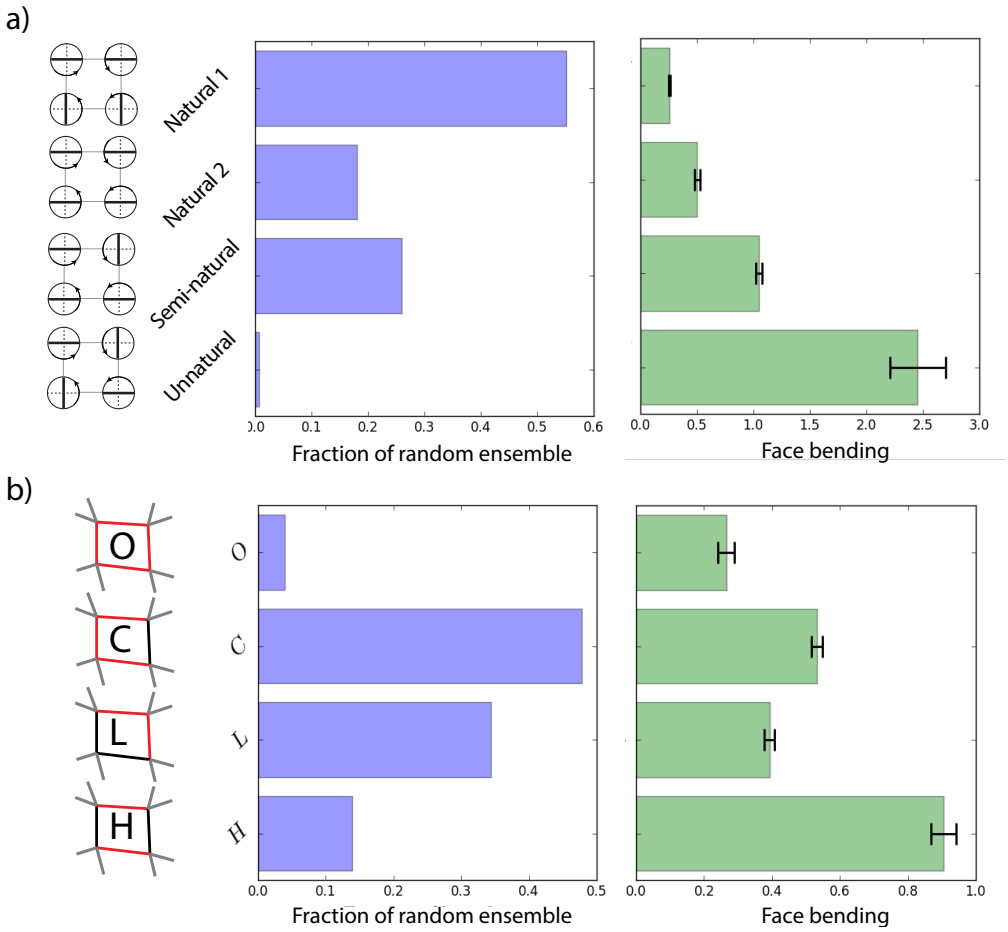

Figure 2.8: Classes of Mountain-Valley choices occur with widely varying frequency and face bending

We generated 1000 random quads by displacing vertices randomly and folded them using random torques. (a) Most of the ensemble folded in the Natural 1 ($\sim$ 55%) or Semi-natural type ($\sim$ 28%) of Mountain-Valley type (blue bars). The average face bending (i.e., stiffness) during folding (green bars) was significantly higher for Unnatural and Semi-natural Mountain-Valley types, as also shown in Fig. 3 of the main paper. (b) We classified the same set of random quads by the type of Mountain or Valley data of the four internal creases. For example, quads of type *O* have internal creases of the same type - the MV states of external legs are ignored in this classification. *C* and *L* type quads are the most common (blue bars) and that *H* and *C* type quads are less foldable than others (green bars). Note that the two kinds of classification shown here in (a) and (b) are independent; e.g., a quad of type, say *O*, could be Natural,Semi-Natural or Unnatural and vice-versa.

2.4 Supplementary Notes

Supplementary Note 1 - Vertex transfer and Loop equations

The basic unit of Origami consists of a quadrivalent vertex, the intersection of four creases. Creases are labeled with Greek letters α,β,γ,δ, where the angles between them (design space) are noted with θ, subscripted by the corresponding adjacent two creases. The angle to which a certain crease folds is given by ρ, subscripted with the corresponding crease label (see Fig. 2.6a).

Any folded configuration of the vertex is given by the four crease folding (or dihedral) angles $\rho_\alpha, \rho_\beta, \rho_\gamma, \rho_\delta$. If we are given any one folding angle, say ρ_α, and the in-plane angles $\theta_{\alpha\beta}, \theta_{\beta\gamma}, \theta_{\gamma\delta}, \theta_{\delta\alpha}$, we can use spherical trigonometry to compute the remaining three fold angles $\rho_\beta, \rho_\gamma, \rho_\delta$.

In the main text, we wrote these equations symbolically in terms of transfer functions $T_{\alpha\beta}$. Here we present implicit formulas for these transfer functions in terms of $\mathrm{C}_\alpha \equiv \cos\rho_\alpha$:

$$\text{Adjacent transfer relation } \alpha \to \beta,\ \rho_\beta = T_{\alpha\beta}(\rho_\alpha;\{\theta\}):$$

$$\begin{aligned}(\mathrm{c}_{\beta\gamma}\mathrm{c}_{\gamma\delta} - \mathrm{s}_{\delta\alpha}\mathrm{s}_{\alpha\beta}\mathrm{C}_\alpha)^2 + (\mathrm{c}_{\gamma\delta}\mathrm{c}_{\delta\alpha} - \mathrm{s}_{\alpha\beta}\mathrm{s}_{\beta\gamma}\mathrm{C}_\beta)^2 - (\mathrm{c}_{\delta\alpha}^2+\mathrm{c}_{\beta\gamma}^2-1)(\mathrm{c}_{\alpha\beta}^2+\mathrm{c}_{\gamma\delta}^2-1) \\ = ((\mathrm{c}_{\delta\alpha}\mathrm{c}_{\alpha\beta} + \mathrm{s}_{\delta\alpha}\mathrm{s}_{\alpha\beta}\mathrm{C}_\alpha)(\mathrm{c}_{\alpha\beta}\mathrm{c}_{\beta\gamma} + \mathrm{s}_{\alpha\beta}\mathrm{s}_{\beta\gamma}\mathrm{C}_\beta) - (\mathrm{c}_{\delta\alpha}\mathrm{c}_{\beta\gamma} + \mathrm{c}_{\alpha\beta}\mathrm{c}_{\gamma\delta}))^2\end{aligned} \tag{2.8}$$

$$\text{Transverse transfer } \alpha \to \gamma,\ \rho_\gamma = T_{\alpha\gamma}(\rho_\alpha;\{\theta\}):$$

$$(\mathrm{c}_{\delta\alpha}\mathrm{c}_{\alpha\beta} + \mathrm{s}_{\delta\alpha}\mathrm{s}_{\alpha\beta}\mathrm{C}_\alpha) = (\mathrm{c}_{\beta\gamma}\mathrm{c}_{\gamma\delta} + \mathrm{s}_{\beta\gamma}\mathrm{s}_{\gamma\delta}\mathrm{C}_\gamma) \tag{2.9}$$

$$\text{Reverse adjacent transfer } \alpha \to \delta,\ \rho_\delta = T_{\alpha\delta}(\rho_\alpha;\{\theta\}):$$

$$\begin{aligned}(\mathrm{c}_{\alpha\beta}\mathrm{c}_{\beta\gamma} - \mathrm{s}_{\gamma\delta}\mathrm{s}_{\delta\alpha}\mathrm{C}_\delta)^2 + (\mathrm{c}_{\beta\gamma}\mathrm{c}_{\gamma\delta} - \mathrm{s}_{\delta\alpha}\mathrm{s}_{\alpha\beta}\mathrm{C}_\alpha)^2 - (\mathrm{c}_{\gamma\delta}^2+\mathrm{c}_{\alpha\beta}^2-1)(\mathrm{c}_{\delta\alpha}^2+\mathrm{c}_{\beta\gamma}^2-1) \\ = ((\mathrm{c}_{\gamma\delta}\mathrm{c}_{\delta\alpha} + \mathrm{s}_{\gamma\delta}\mathrm{s}_{\delta\alpha}\mathrm{C}_\delta)(\mathrm{c}_{\delta\alpha}\mathrm{c}_{\alpha\beta} + \mathrm{s}_{\delta\alpha}\mathrm{s}_{\alpha\beta}\mathrm{C}_\alpha) - (\mathrm{c}_{\gamma\delta}\mathrm{c}_{\alpha\beta} + \mathrm{c}_{\delta\alpha}\mathrm{c}_{\beta\gamma}))^2\end{aligned} \tag{2.10}$$

where $s_{\alpha\beta} \equiv \sin(\theta_{\alpha\beta}), c_{\alpha\beta} \equiv \cos(\theta_{\alpha\beta}), C_\alpha \equiv \cos(\rho_\alpha)$ and similarly for all other factors. Note that, in these equations, we have adopted a convention in which at the flat unfolded state $\rho = \pi$ and hence $C = -1$.

Linearization and branches: We will primarily use the more complex transverse transfer function $T_{\alpha\beta}$ between two consecutive creases. The transfer function can be linearized about the flat state, $\rho_\alpha = \pi - \epsilon\rho_\alpha$, to find

$$\epsilon\rho_\beta = R_{\alpha\beta}\,\epsilon\rho_\alpha, \text{ with } \boxed{R^2_{\alpha\beta} = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A},} \tag{2.11}$$

with

$$A \equiv (s_{\alpha\beta}s_{\beta\gamma})^2(1 - (c_{\delta\alpha}c_{\alpha\beta} - s_{\delta\alpha}s_{\alpha\beta})^2) \tag{2.12}$$

$$B \equiv 2(s_{\delta\alpha}s_{\alpha\beta})(s_{\alpha\beta}s_{\beta\gamma})((c_{\delta\alpha}c_{\beta\gamma} + c_{\alpha\beta}c_{\gamma\delta}) - 2(c_{\delta\alpha}c_{\alpha\beta} - s_{\delta\alpha}s_{\alpha\beta})(c_{\alpha\beta}c_{\beta\gamma} - s_{\alpha\beta}s_{\beta\gamma})) \tag{2.13}$$

$$C \equiv (s_{\delta\alpha}s_{\alpha\beta})^2(1 - (c_{\alpha\beta}c_{\beta\gamma} - s_{\alpha\beta}s_{\beta\gamma})^2) \tag{2.14}$$

Note that we have two distinct solutions since Equation (2.9) is quadratic in $\cos\rho_\alpha, \cos\rho_\beta$. This choice arises because a single vertex has two distinct folding modes [12]. We pick between these two branches by setting one of the two creases whose sum of angles around it are less than π (see Fig. 2.6a), to be the odd-one-out in the Mountain/Valley pattern. For the vertex shown here, the two options for the odd-one-out crease are highlighted in orange. The larger value of R should be used when ρ_α and ρ_γ have opposite signs (i.e., are of opposite Mountain-Valley state). One intuitive explanation is that if the angles at the vertex were all $\frac{\pi}{2}$, R would be infinite if ρ_α, ρ_γ have opposite signs. The choice of branch for each vertex around the quad sets the MV class of the quad, as discussed in the main text.

Loop equation around a quadrilateral

Strategy

Now consider a quadrilateral made of four general vertices (Fig. 2.6b). At each vertex a, we write the transfer equation between creases forming the sides of the quadrilateral, $\rho_i = T_a(\rho_{i+1})$.

We may rewrite this equation as $\mathbf{z}_a \equiv \rho_i - T_a(\rho_{i+1}) = 0$. As the quantity $\mathbf{z}_a$ vanishes for each vertex of the quadrilateral, we define $\mathbf{z} \equiv \sum_a \mathbf{z}_a = 0$. The flat state trivially has $\mathbf{z} = 0$, as all $\rho = 0$; we want to find a continuous path $\rho_i(t)$ in folding angle space, parameterized by some variable t, beginning at the flat state and maintaining $\mathbf{z} = 0$.

In the spirit of perturbation theory we compute successive derivatives of $\mathbf{z}$ with respect to t, in order to set them to zero at $t = 0$:

$$\begin{aligned} \dot{\mathbf{z}}|_{t=0} &= 0 \\ \ddot{\mathbf{z}}|_{t=0} &= 0 \\ \dddot{\mathbf{z}}|_{t=0} &= 0 \\ &\vdots \end{aligned} \tag{2.15}$$

Henceforth we omit the designation $|_{t=0}$ as all derivatives are evaluated at $t = 0$.

Since at each order, four new variables enter (higher t derivatives of each $\mathbf{C}_i \equiv \cos(\rho_i)$), and four new quantities must be set to zero (the four components of the appropriate t derivative of $\mathbf{z}$), we might expect that there is exactly one solution for $\{\dot{\mathbf{C}}_i, \ddot{\mathbf{C}}_i, ...\}$ and our task is simply to find it. However, that solution is the trivial $\mathbf{C}_i(t) = 0$ for all t. In order to have a non-trivial solution, we must make a matrix corresponding to the highest derivatives singular: see below. But since these matrices appears in the equation at every order, solving each additional order requires the imposition of one more constraint.

Zeroth loop equation

The first equation is trivially satisfied for any path $\rho_i(t)$; to see this, note that

$$\dot{\mathbf{z}} = \frac{\partial \mathbf{z}}{\partial \mathbf{C}_i}\dot{\mathbf{C}}_i \tag{2.16}$$

where repeated indices are summed over.

Here it is worthwhile to note the two different 4-dimensional vector spaces with which we are working. $\mathbf{C}_i$ has $i = 1, 2, 3, 4$ running over the internal edges of the quad pattern shown above, while the four components of $\mathbf{z}$ are associated with the four vertices. Thus $\frac{\partial \mathbf{z}_a}{\partial \mathbf{C}_i}$ is a matrix operator from edge space to corner space, and higher derivatives are higher order tensor operators. Of course, the a, i component of this operator will be zero if edge i does not join vertex a.

Now from Eqn. 2.9, $\frac{\partial \mathbf{z}_a}{\partial \mathbf{C}_i}$ turns out to be zero at $\mathrm{C}_\alpha = \mathrm{C}_\beta = -1$ if the sum of the four in-plane angles at vertex a is 2π, as in the case we are considering. So any set of $\dot{\mathbf{C}}_i$s will satisfy this condition and we do not have any non-trivial constraint.

First loop equation

The first non-trivial condition is at second order (repeated indices summed over),

$$\ddot{\mathbf{z}} = \frac{\partial \mathbf{z}}{\partial \mathbf{C}_i}\ddot{\mathbf{C}}_i + \frac{\partial^2 \mathbf{z}}{\partial \mathbf{C}_i \partial \mathbf{C}_j}\dot{\mathbf{C}}_i\dot{\mathbf{C}}_j = \frac{\partial^2 \mathbf{z}}{\partial \mathbf{C}_i \partial \mathbf{C}_j}\dot{\mathbf{C}}_i\dot{\mathbf{C}}_j \tag{2.17}$$

Working out $\frac{\partial^2 \mathbf{z}}{\partial \mathbf{C}_i \partial \mathbf{C}_j}$, we find that $\ddot{\mathbf{z}}_a$ will be zero if $\dot{\mathbf{C}}_{a+1} = R_a \dot{\mathbf{C}}_a$ (where again R_a is the linear transfer coefficient between creases i and $i+1$). In other words, the equation above will be satisfied together if $\dot{\mathbf{C}}$ is a zero eigenvector of the matrix,

$$\begin{bmatrix} R_1 & -1 & 0 & 0 \\ 0 & R_2 & -1 & 0 \\ 0 & 0 & R_3 & -1 \\ -1 & 0 & 0 & R_4 \end{bmatrix}. \tag{2.18}$$

The first loop equation is thus obtained by imposing that the above matrix is singular (so that it has a non-trivial zero eigenvector):

$$\boxed{R_1 R_2 R_3 R_4 = 1} \tag{2.19}$$

where the explicit form of R is given in equation (2.11). If the first loop equation is satisfied, the solution $\dot{\mathbf{C}}$ that satisfies equation (2.17) obeys

$$\dot{\mathbf{C}}_{i+1} = R_i \dot{\mathbf{C}}_i, \tag{2.20}$$

For later use, we take $\dot{\mathbf{C}}$ to be normalized to have magnitude 1.

Second loop equation

Now, the third order derivative is

$$\dddot{\mathbf{z}} = \frac{\partial \mathbf{z}}{\partial \mathbf{C}_i} \dddot{\mathbf{C}}_i + 3 \frac{\partial^2 \mathbf{z}}{\partial \mathbf{C}_i \partial \mathbf{C}_j} \ddot{\mathbf{C}}_i \dot{\mathbf{C}}_j + \frac{\partial^3 \mathbf{z}}{\partial \mathbf{C}_i \partial \mathbf{C}_j \partial \mathbf{C}_k} \dot{\mathbf{C}}_i \dot{\mathbf{C}}_j \dot{\mathbf{C}}_k \tag{2.21}$$

$$= 3 \frac{\partial^2 \mathbf{z}}{\partial \mathbf{C}_i \partial \mathbf{C}_j} \ddot{\mathbf{C}}_i \dot{\mathbf{C}}_j + \frac{\partial^3 \mathbf{z}}{\partial \mathbf{C}_i \partial \mathbf{C}_j \partial \mathbf{C}_k} \dot{\mathbf{C}}_i \dot{\mathbf{C}}_j \dot{\mathbf{C}}_k \tag{2.22}$$

$$\equiv 3M\ddot{\mathbf{C}} + \mathbf{v_{111}}. \tag{2.23}$$

Here we have made the definition $\mathbf{v_{111}} = \frac{\partial^3 \mathbf{z}}{\partial \mathbf{C}_i \partial \mathbf{C}_j \partial \mathbf{C}_k} \dot{\mathbf{C}}_i \dot{\mathbf{C}}_j \dot{\mathbf{C}}_k$, intended as a special case of the general definition

$$\mathbf{v_{p_1 p_2 \cdots p_n}} = \frac{\partial^n \mathbf{z}}{\partial \mathbf{C}_{i_1} \partial \mathbf{C}_{i_2} \ldots \partial \mathbf{C}_{i_n}} \frac{d^{p_1} \mathbf{C}_{i_1}}{dt^{p_1}} \frac{d^{p_2} \mathbf{C}_{i_2}}{dt^{p_2}} \ldots \frac{d^{p_n} \mathbf{C}_{i_n}}{dt^{p_n}}. \tag{2.24}$$

We have also defined the matrix $M = \frac{\partial^2 \mathbf{z}}{\partial \mathbf{C}_i \partial \mathbf{C}_j} \dot{\mathbf{C}}_j$. Note that M has one vertex index (from $\mathbf{z}$) and one edge index i (from $\mathbf{C}_i$). Since $\dot{\mathbf{C}}$ is known, M and $\mathbf{v_{111}}$ are explicitly known: they can be written as

$$M = \begin{bmatrix} D_1\dot{\mathbf{C}}_2 & -D_1\dot{\mathbf{C}}_1 & 0 & 0 \\ 0 & D_2\dot{\mathbf{C}}_3 & -D_2\dot{\mathbf{C}}_2 & 0 \\ 0 & 0 & D_3\dot{\mathbf{C}}_4 & -D_3\dot{\mathbf{C}}_3 \\ -D_4\dot{\mathbf{C}}_4 & 0 & 0 & D_4\dot{\mathbf{C}}_1 \end{bmatrix}, \quad \text{and} \quad \mathbf{v_{111}} = -3\begin{bmatrix} D_1\dot{\mathbf{C}}_1\dot{\mathbf{C}}_2K_1 \\ D_2\dot{\mathbf{C}}_2\dot{\mathbf{C}}_3K_2 \\ D_3\dot{\mathbf{C}}_3\dot{\mathbf{C}}_4K_3 \\ D_4\dot{\mathbf{C}}_4\dot{\mathbf{C}}_1K_4 \end{bmatrix} \tag{2.25}$$

with D, E, F and K define at a given vertex by

$$\boxed{K_i = \frac{\dot{\mathbf{C}}_i E + \dot{\mathbf{C}}_{i+1} F}{D},} \tag{2.26}$$

$$\text{with } D = \frac{C}{R} + \frac{B}{2}, \quad E = 4s_{\alpha\beta}^3 s_{\beta\gamma}(c_{\alpha\beta}c_{\beta\gamma} - s_{\alpha\beta}s_{\beta\gamma})s_{\delta\alpha}^2, \quad F = 4s_{\alpha\beta}^3 s_{\beta\gamma}^2 s_{\delta\alpha}(c_{\alpha\beta}c_{\delta\alpha} - s_{\alpha\beta}s_{\delta\alpha}). \tag{2.27}$$

We know from equation (2.17) that M is singular. Hence, if we want $\dddot{\mathbf{z}}|_{t=0} = 0$, we need to ensure that $\mathbf{v_{111}}$ is in the range of M.

The condition for this to lie in the range of M is that

$$\boxed{K_1 + K_2 + K_3 + K_4 = 0}. \tag{2.28}$$

This can be seen in various ways, including by making the ansatz

$$\ddot{\mathbf{C}} = \begin{bmatrix} \dot{\mathbf{C}}_1(\dot{\mathbf{C}}_2^2K_1 + \dot{\mathbf{C}}_3^2(K_1 + K_2) + \dot{\mathbf{C}}_4^2(K_1 + K_2 + K_3)) \\ \dot{\mathbf{C}}_2(\dot{\mathbf{C}}_3^2K_2 + \dot{\mathbf{C}}_4^2(K_2 + K_3) + \dot{\mathbf{C}}_1^2(K_2 + K_3 + K_4)) \\ \dot{\mathbf{C}}_3(\dot{\mathbf{C}}_4^2K_3 + \dot{\mathbf{C}}_1^2(K_3 + K_4) + \dot{\mathbf{C}}_2^2(K_3 + K_4 + K_1)) \\ \dot{\mathbf{C}}_4(\dot{\mathbf{C}}_1^2K_4 + \dot{\mathbf{C}}_2^2(K_4 + K_1) + \dot{\mathbf{C}}_3^2(K_4 + K_1 + K_2)) \end{bmatrix}. \tag{2.29}$$

We then have

$$3M\ddot{\mathbf{C}} = 3\begin{bmatrix} D_1\dot{\mathbf{C}}_1\dot{\mathbf{C}}_2((\dot{\mathbf{C}}_2^2+\dot{\mathbf{C}}_3^2+\dot{\mathbf{C}}_4^2)K_1 - \dot{\mathbf{C}}_1^2(K_2+K_3+K_4)) \\ D_2\dot{\mathbf{C}}_2\dot{\mathbf{C}}_3((\dot{\mathbf{C}}_3^2+\dot{\mathbf{C}}_4^2+\dot{\mathbf{C}}_1^2)K_2 - \dot{\mathbf{C}}_2^2(K_3+K_4+K_1)) \\ D_3\dot{\mathbf{C}}_3\dot{\mathbf{C}}_4((\dot{\mathbf{C}}_4^2+\dot{\mathbf{C}}_1^2+\dot{\mathbf{C}}_2^2)K_3 - \dot{\mathbf{C}}_3^2(K_4+K_1+K_2)) \\ D_4\dot{\mathbf{C}}_4\dot{\mathbf{C}}_1((\dot{\mathbf{C}}_1^2+\dot{\mathbf{C}}_2^2+\dot{\mathbf{C}}_3^2)K_4 - \dot{\mathbf{C}}_4^2(K_1+K_2+K_3)) \end{bmatrix}, \tag{2.30}$$

which is equal to $\mathbf{v_{111}}$ iff $K_1 + K_2 + K_3 + K_4 = 0$.

Third loop equation

The fourth order derivative is

$$\begin{aligned} \dddot{\mathbf{z}} &= \frac{\partial \mathbf{z}}{\partial \mathbf{C}_i}\dddot{\mathbf{C}}_i + 4\frac{\partial^2 \mathbf{z}}{\partial \mathbf{C}_i \partial \mathbf{C}_j}\ddot{\mathbf{C}}_i\dot{\mathbf{C}}_j + 3\frac{\partial^2 \mathbf{z}}{\partial \mathbf{C}_i \partial \mathbf{C}_j}\ddot{\mathbf{C}}_i\ddot{\mathbf{C}}_j \\ &\quad + 6\frac{\partial^3 \mathbf{z}}{\partial \mathbf{C}_i \partial \mathbf{C}_j \partial \mathbf{C}_k}\ddot{\mathbf{C}}_i\dot{\mathbf{C}}_j\dot{\mathbf{C}}_k + \frac{\partial^4 \mathbf{z}}{\partial \mathbf{C}_i \partial \mathbf{C}_j \partial \mathbf{C}_k \partial \mathbf{C}_l}\dot{\mathbf{C}}_i\dot{\mathbf{C}}_j\dot{\mathbf{C}}_k\dot{\mathbf{C}}_l \\ &= 4M\dddot{\mathbf{C}} + 3\mathbf{v_{22}} + 6\mathbf{v_{211}} + \mathbf{v_{1111}} = 0. \end{aligned} \tag{2.31}$$

Once again, we can evaluate the vector terms explicitly:

$$3\mathbf{v_{22}} = 3\begin{bmatrix} C_1\ddot{\mathbf{C}}_1^2 + A_1\ddot{\mathbf{C}}_2^2 + B_1\ddot{\mathbf{C}}_1\ddot{\mathbf{C}}_2 \\ C_2\ddot{\mathbf{C}}_2^2 + A_2\ddot{\mathbf{C}}_3^2 + B_2\ddot{\mathbf{C}}_2\ddot{\mathbf{C}}_3 \\ C_3\ddot{\mathbf{C}}_3^2 + A_3\ddot{\mathbf{C}}_4^2 + B_3\ddot{\mathbf{C}}_3\ddot{\mathbf{C}}_4 \\ C_4\ddot{\mathbf{C}}_4^2 + A_4\ddot{\mathbf{C}}_1^2 + B_4\ddot{\mathbf{C}}_4\ddot{\mathbf{C}}_1 \end{bmatrix}, \quad \mathbf{v_{1111}} = 6\begin{bmatrix} G_1\dot{\mathbf{C}}_1^2\dot{\mathbf{C}}_2^2 \\ G_2\dot{\mathbf{C}}_2^2\dot{\mathbf{C}}_3^2 \\ G_3\dot{\mathbf{C}}_3^2\dot{\mathbf{C}}_4^2 \\ G_4\dot{\mathbf{C}}_4^2\dot{\mathbf{C}}_1^2 \end{bmatrix}$$

$$6\mathbf{v_{211}} = -6\begin{bmatrix} E_1(2\ddot{\mathbf{C}}_1\dot{\mathbf{C}}_1\dot{\mathbf{C}}_2 + \ddot{\mathbf{C}}_2\dot{\mathbf{C}}_1^2) + F_1(2\ddot{\mathbf{C}}_2\dot{\mathbf{C}}_1\dot{\mathbf{C}}_2 + \ddot{\mathbf{C}}_1\dot{\mathbf{C}}_2^2) \\ E_2(2\ddot{\mathbf{C}}_2\dot{\mathbf{C}}_2\dot{\mathbf{C}}_3 + \ddot{\mathbf{C}}_3\dot{\mathbf{C}}_2^2) + F_2(2\ddot{\mathbf{C}}_3\dot{\mathbf{C}}_2\dot{\mathbf{C}}_3 + \ddot{\mathbf{C}}_2\dot{\mathbf{C}}_3^2) \\ E_3(2\ddot{\mathbf{C}}_3\dot{\mathbf{C}}_3\dot{\mathbf{C}}_4 + \ddot{\mathbf{C}}_4\dot{\mathbf{C}}_3^2) + F_3(2\ddot{\mathbf{C}}_4\dot{\mathbf{C}}_3\dot{\mathbf{C}}_4 + \ddot{\mathbf{C}}_3\dot{\mathbf{C}}_4^2) \\ E_4(2\ddot{\mathbf{C}}_4\dot{\mathbf{C}}_4\dot{\mathbf{C}}_1 + \ddot{\mathbf{C}}_1\dot{\mathbf{C}}_4^2) + F_4(2\ddot{\mathbf{C}}_1\dot{\mathbf{C}}_4\dot{\mathbf{C}}_1 + \ddot{\mathbf{C}}_4\dot{\mathbf{C}}_1^2) \end{bmatrix} \tag{2.32}$$

where $G = -4s^4_{\alpha\beta}s^2_{\beta\gamma}s^2_{\delta\alpha}$. Let us define L_i by setting,

$$3\mathbf{v_{22}} + 6\mathbf{v_{211}} + \mathbf{v_{1111}} \equiv \begin{bmatrix} \dot{\mathbf{C}}_1\dot{\mathbf{C}}_2 D_1 L_1 \\ \dot{\mathbf{C}}_2\dot{\mathbf{C}}_3 D_2 L_2 \\ \dot{\mathbf{C}}_3\dot{\mathbf{C}}_4 D_3 L_3 \\ \dot{\mathbf{C}}_4\dot{\mathbf{C}}_1 D_4 L_4 \end{bmatrix} \tag{2.33}$$

By comparison with equations (2.26) and (2.28), we see that there will be a solution if we can write with

$$\boxed{L_1 + L_2 + L_3 + L_4 = 0}. \tag{2.34}$$

We can work out the explicit form of L_i from the above equations,

$$\begin{aligned} D_iL_i =& 3(\frac{C_i}{R_i}\Gamma_i^2 + A_iR_i\Gamma_{i+1}^2 + B_i\Gamma_i\Gamma_{i+1}) \\ & - 6\left(E_i(2\dot{\mathbf{C}}_i\Gamma_i + \Gamma_{i+1}\dot{\mathbf{C}}_i) + F_i(2\dot{\mathbf{C}}_{i+1}\Gamma_{i+1} + \dot{\mathbf{C}}_{i+1}\Gamma_i)\right) + 6G_i\dot{\mathbf{C}}_i\dot{\mathbf{C}}_{i+1}, \end{aligned} \tag{2.35}$$

$$\text{with } \Gamma_i = \dot{\mathbf{C}}_{i+1}^2K_i + \dot{\mathbf{C}}_{i+2}^2(K_i + K_{i+1}) + \dot{\mathbf{C}}_{i+3}^2(K_i + K_{i+1} + K_{i+2}). \tag{2.36}$$

Note that here and hereafter, subscript indices $i + 1$, $i + 2$ and $i + 3$ must be understood as modulo 4. Since $\Gamma_{i+1} = \Gamma_i - K_i$, we can write

$$\begin{aligned} D_iL_i =& 3(\frac{C_i}{R_i}\Gamma_i^2 + A_iR_i(\Gamma_i - K_i)^2 + B_i\Gamma_i(\Gamma_i - K_i)) \\ & - 6(E_i(2\dot{\mathbf{C}}_i\Gamma_i + (\Gamma_i - K_i)\dot{\mathbf{C}}_i) + F_i(2\dot{\mathbf{C}}_{i+1}(\Gamma_i - K_i) + \dot{\mathbf{C}}_{i+1}\Gamma_i)) + 6G_i\dot{\mathbf{C}}_i\dot{\mathbf{C}}_{i+1}. \end{aligned}$$

Using the definitions of R, K and D this further simplifies

$$\boxed{L_i = -12K_i\Gamma_i + \frac{3A_iR_iK_i^2}{D_i} + 6K_i^2 + 6\frac{F_iK_i}{D_i}\dot{\mathbf{C}}_{i+1} + 6\frac{G_i}{D_i}\dot{\mathbf{C}}_i\dot{\mathbf{C}}_{i+1}}. \tag{2.37}$$

Generating arbitrary loop equations

The nth order derivative of $\mathbf{z}$ has two types of term. The first is the single term $nM\frac{d^{n-1}\mathbf{C}}{dt^{n-1}}$. The second is terms of the form $\mathbf{v_{p_1(n-p_1)}}$, $\mathbf{v_{p_1p_2(n-p_1-p_2)}}$ and $\mathbf{v_{p_1p_2p_3(n-p_1-p_2-p_3)}}$, with numerical factors that can be straightforwardly determined by combinatorics. Importantly, terms of this form can have at most 4 indices, because the equation for $\mathbf{z}$ is a fourth order polynomial in the $\mathbf{C}_i$s. Furthermore, the highest t derivative of a $\mathbf{C}_i$ that appears in these terms is $n-2$, and the first $n-2$ derivatives of $\mathbf{C}$ were calculated using the equations arising from the first $n-1$ derivatives of $\mathbf{z}$. Thus the equation arising from setting the nth derivative of $\mathbf{z}$ to zero is

$$nM\frac{d^{n-1}\mathbf{C}}{dt^{n-1}} + \mathbf{V^n} = 0, \tag{2.38}$$

where $\mathbf{V^n}$ is the sum of known vectors. As in the derivation of the second and third loop equations, we then define $\mathbf{L^n}$ by

$$V_i^n = \dot{\mathbf{C}}_i\dot{\mathbf{C}}_{i+1}D_iL_i^n, \tag{2.39}$$

and the $n-1$th loop equation is $L_1^n + L_2^n + L_3^n + L_4^n = 0$. Finally,

$$\frac{d^{n-1}\mathbf{C}_i}{dt^{n-1}} = \dot{\mathbf{C}}_i(\dot{\mathbf{C}}_{i+1}^2L_i^n + \dot{\mathbf{C}}_{i+2}^2(L_i^n + L_{i+1}^n) + \dot{\mathbf{C}}_{i+3}^2(L_i^n + L_{i+1}^n + L_{i+2}^n)). \tag{2.40}$$

MATLAB code to generate such equations is provided as supplementary material.

Supplementary Note 2 - Flat foldability, rigid foldability and the loop equations

Flat-foldability

A flat foldable pattern is an origami pattern that can be folded completely flat down to a plane. Kawasaki-Justin showed that a necessary condition for a single 4-vertex to be flat-foldable is that opposing in-plane angles at the vertex sum to π. (Bern and Hayes [46] showed that no simple global criterion exists to check if a pattern is globally flat foldable, i.e., avoids global collisions while folding flat.)

Despite the name, flat foldability of vertices does not imply any foldability of a crease pattern made of such vertices. In fact, such a crease pattern may not be foldable even to first order! If one would like to reach the flat-folded state of a generic crease pattern with flat foldable 4-vertices, the system would have to be "snapped" through states with large face bending, requiring energy input.

Flat-foldable vertices are more restricted than general vertices in the number of design parameters available and the folded geometries they can acquire. Flat-foldable vertices have only two independent design parameters (i.e., in-plane angles) at each vertex since opposite angles need to sum to π; in contrast, general vertices studied in this paper have three independent in-plane angles. Further, the folded geometries that a flat-foldable vertex can acquire is limited; the four folding angles $\rho_1, \rho_2, \rho_3, \rho_4$ at a vertex are constrained such that $|\rho_1| = |\rho_3|, |\rho_2| = |\rho_4|$. That is, opposite creases are forced to fold to the same extent. In contrast, general vertices can achieve arbitrary folding angles $\rho_1, \rho_2, \rho_3, \rho_4$, allowing more kinds of local geometry.

Tachi's loop equation for flat-foldable quads

Tachi identified a single loop equation as a necessary and sufficient condition for quads made of flat foldable vertices to also be rigidly foldable with no face bending at all to any order [36]. Tachi later derived a 1-st order loop equation for general non-flat foldable quads [43]. In contrast, we studied general quads with possibly non-flat foldable vertices and found that general patterns are faced with a hierarchy of loop equations. Solving these in sequence produces a more and more

foldable pattern over orders of magnitude.

Our results for tunable foldability of general crease patterns with arbitrary Mountain-Valley choice relate to Tachi's work on rigid foldability of crease patterns with 'flat-foldable' vertices. For quads with flat foldable vertices, the first of our loop equations, $\Pi R = 1$ is the only independent equation. Solving it immediately solves all other equations. Thus, for quads with flat foldable vertices, first order foldability implies rigid foldability. This is consistent with Tachi's result that there is only one loop equation for quads with flat foldable vertices. We have also numerically verified that quads with flat foldable vertices satisfying Tachi's loop equation automatically satisfy all of our loop equations.

The amount of rigid foldable patterns

To be rigidly foldable with no face bending, the in-plane angles of a quad must solve our infinite hierarchy of loop equations. Since a quad has only 11 independent design parameters (i.e., in-plane angles θ), no rigidly foldable quads can be expected to exist unless the loop equations are somehow not independent.

We verified that the first **five** of our loop equations are indeed independent; for example, as quads solve the fourth loop equation $\Sigma M = 0$ more and more precisely, the residue of the fifth equation ΣN stays finite. Thus, the space of rigidly foldable quad patterns is at most $11 - 5 = 6$ parameter. Combined with Tachi's results showing a 6 parameter space of rigidly foldable quads, this indicates that higher loop equations, beyond the 5^{th} loop equation, are not actually independent. (To see this, note that 'flat foldable' quads studied by Tachi [36] have 7 independent design parameters. On imposing Tachi's single loop equation needed for rigid foldability, one obtains a 6 parameter family of rigidly foldable quads.)

To summarize, our work shows that space of rigidly foldable quads is no more than a 6 parameter family and Tachi's earlier work [36] already demonstrated the existence of a 6 parameter rigidly foldable family.

Other rigidly foldable patterns:

In principle, there could exist other 6 parameter (or lower) families of rigidly foldable quads

that are not flat foldable and hence not in Tachi's family. Indeed, non-flat foldable but rigidly foldable quads can easily exist due to symmetry. For example, the trapezoid-like crease pattern (Fig. 2.6c) shows a member of a 5-parameter family of quads that is rigidly foldable due to symmetry. The reflection symmetry in these trapezoids means that the transfer functions cancel to all orders in pairs of adjacent vertices. Besides such solutions due to symmetry, we were not able to find any rigidly foldable non-flat foldable quads by sampling random quads and solving loop equations by gradient descent. While such numeric searches cannot mathematically rule out the possibility of rigidly foldable quads (besides those described by Tachi) that are solutions of all our loop equations, any such rigidly foldable quads cannot constitute more than a 6-parameter family.

2.5 Supplementary Methods

Manufacturing origami prototypes

We made origami prototypes by cutting 120 lb cardstock using a laser cutter. Patterns were roughly of size 10 cm x 10 cm. Creases were created using a perforation pattern of 0.6 mm cuts with 0.7 mm gaps. These patterns were folded by hand according to their respective designed MV configurations. Prototypes in Fig. 2.1b and 2.1e were recolored using Adobe Photoshop to ensure distinct colors in Fig. 2.1.

Folding using constraint matrices

We simulate the folding of rigid origami using the constraint matrices introduced by Tachi [40]. For each vertex with edges e_i, in-plane angles θ_{ij} and creases folded to a general set of fold angles ρ_i. Define $A_{ij} = R(\theta_{ij}, n_{ij})$, the $SO(3)$ rotation matrix by θ_{ij} degrees about the normal n_{ij} to the face between edge i and j (i.e., $n_{ij} = e_i \times e_j$ where e_i and e_j are unit vectors along the edges). Let $B_i = R(\rho_i, e_i)$ be the rotation matrix about edge i by an angle ρ_i. If the vertex is folded without any tearing, gaps or bending of faces between edges, the product of these rotation matrices should

bring us back to the origin; $F(\{\rho_i\}) = A_{12}B_2A_{23}B_3A_{34}B_4A_{41}B_1 = \mathbf{I}$. If a small change in fold angles $\delta\rho$ does not violate these constraints, it must satisfy, $\delta F = \sum_i \frac{\partial F}{\partial \rho_i}\delta\rho_i = 0$. Note that F is an $SO(3)$ matrix and hence δF is a skew-symmetric $so(3)$ matrix. Hence we can define $C_{ai} = \frac{\epsilon_{abc}\delta F_{bc}}{\delta\rho_i}$ which allows us to re-write the constraint equation as $C(\rho)\delta\rho = 0$.

Thus, the allowed motions $\delta\rho$ at any given folded state ρ are given by the zero modes of $C(\rho)$. Hence we solve the differential equation $\dot{\rho} = (\mathbf{I} - C^+C)\tau_{\text{app}}$ where τ_{app} is set of folding torques applied to creases and C^+ is the pseudo-inverse of C. Here $\mathbf{I} - C^+C$ projects the change of angles caused by τ_{app} to the null space of C. We solve the above equation using MATLAB's ODE solver.

Sampling random simple / augmented quads

A simple quad is uniquely defined by specifying the coordinates of 12 points on a plane. Provided the points are connected through non-intersecting lines, the resulting graph will describe a simple quad (Fig. 2.7d). The same set of points also uniquely define the augmented quad, with the convention that the center diagonal is drawn from the bottom left to the top right (Fig. 2.7d).

Each simple quad is sampled by first setting the 12 points to define a perfect square lattice of unit length 1. Then, both coordinates of each point are altered randomly by adding a number drawn out of a uniform distribution on $[-0.5, 0.5]$. The resulting quad is guaranteed to have a convex quadrilateral at its center, so the corresponding augmented quad can be constructed using the same coordinate set.

A large mesh is sampled similarly, using a larger square lattice as the initial state, e.g., a 2×2 quads mesh starts with a 4×4 square lattice. Lattice points are then displaced using a random uniform distribution. Finally, the center face of each quad is augmented with a diagonal crease.

Detection of folding modes

For each augmented quad (or mesh) sampled by the method described above, we obtain a list of all zero energy modes by applying random normalized torques to the flat state, and folding it using the constraint matrix method. As the simple quad is made of 12 creases, the torque on the

quad is a 12-component vector, one value for every crease. After sampling a number of random torques and recording the resulting folding modes, we sample an additional number of torques and record more unique modes. This step is repeated as long as new unique zero modes are found. By this procedure (nearly) all zero modes are detected, barring those corresponding to small surface areas on the 12-sphere. For larger quad meshes, we carry out the same procedure but with torques vectors of length corresponding to the number of creases in the pattern.

Foldability along a folding mode

Sampled augmented quads were analyzed to find their zero modes, in particular those that are compatible with the corresponding simple quads. However, the existence of a zero mode informs us little about how it behaves when folding the quad. In particular, it is known there exist quads that are bistable (having stable configurations divided by modes that are not precise zero modes). In our setting, bistable modes would have face folding grow up to some angle and then overturn. Such behavior would seem to indicate that the more folded configuration costs less energy.

We study a few of the sampled quads and analyze all of their legal modes (those compatible with the corresponding simple quad). To improve numerical stability, simulation is started in the folded state where the maximum crease angle is 1 Rad. The pattern is then folded backwards toward the flat states at intervals of 0.05 Rad, and forward towards π using the same interval. We note that patterns are folded until the maximum crease angle reaches 3.1 Rad. Folding very close to angle π is avoided, as this angle value indicates a collision of faces, after which the system leaves the domain of validity of our analysis.

Chapter 3

The complexity of folding self-folding origami

Single degree of freedom mechanical structures are attractive in a range of fields as almost any force will actuate that specific designed mode. Much like an umbrella or a folding chair, such 'self-folding' structures can be reliably deployed even in uncertain environments with unreliable actuation forces. This principle has found wide use in kinetic or deployable architecture, heart stents, MEMS, sensors and robots on a range of length scales [10, 26, 27, 60]; recently, self-folding origami has become a popular framework for such applications [14, 31, 34, 36, 45, 51].

The self-folding approach is similar in spirit to other bottom-up methods such as self-assembly of particles [61] and self-folding of polymers [62]; these methods exploit careful programming of interactions to allow for careless actuation at deployment. However, in these other self-actuating frameworks, the interactions needed for the desired assembly or folding inevitably create many other 'distractor' states (e.g., kinetic traps in self-assembly [63, 64, 65] or in protein folding [62, 66, 67]), necessitating more care at deployment than one would naively expect.

Here, we show that it is difficult to fold self-folding origami (a thin sheet pre-creased to allow only a single folding motion) because of a similar inevitable proliferation of distractor folding

branches. The distractor branches, shown schematically in Fig. 3.1, meet at a bifurcation at the flat state but are dead-ends since they are of zero energy only to linear order. The number of distractors grows exponentially with size of the sheet and consequently, most spatial distributions of folding forces will actuate a distractor (Fig. 3.1c,d). As a result, despite having only one extended degree of freedom, self-folding crease patterns require multiple actuators placed at carefully chosen spatial locations for successful actuation.

We trace the origin of distractors to frustrated loops of vertices, each of which can fold along one of two branches. Such frustrated loops create a glassy energy landscape for the sheet around the flat state, i.e., a landscape with an exponential number of local minima corresponding to the distractors. Material properties are expected to modify the precise details of this landscape, yet will not change its fundamental glassy nature. Successful folding must be seeded by actuation at a carefully chosen set of creases that picks out the ground state of the glassy landscape, much like with protein folding [67, 68, 69, 70] and other satisfiability problems [71]. We find that the spatial arrangement of actuators needed can be understood heuristically in terms of unfrustrated 'folding islands', the largest sub-pattern containing a given actuated crease that will fold when cut out of the full pattern.

In this way, our work shows fundamental limits to the programmability of self-folding sheets due to an inevitable glassy landscape of undesired states. In conjunction with similar limits in other bottom up approaches like self-assembly [61] and self-folding polymers [62], our work adds to a common picture of glassiness intrinsic to bottom-up methods based on frustrated and disordered interactions, independent of the details of specific implementations.

In addition, our results provide a practical means of understanding where to place active creases; e.g., in hydrogels or shape memory alloys, one must choose the active hinges; our theory predicts which combination of hinges would be successful and even predicts that sometimes, adding a new active crease (aiding in the right direction) to an existing successful actuation can in fact prevent folding.

Our results on glassiness and the difficulty of physically folding origami superficially resem-

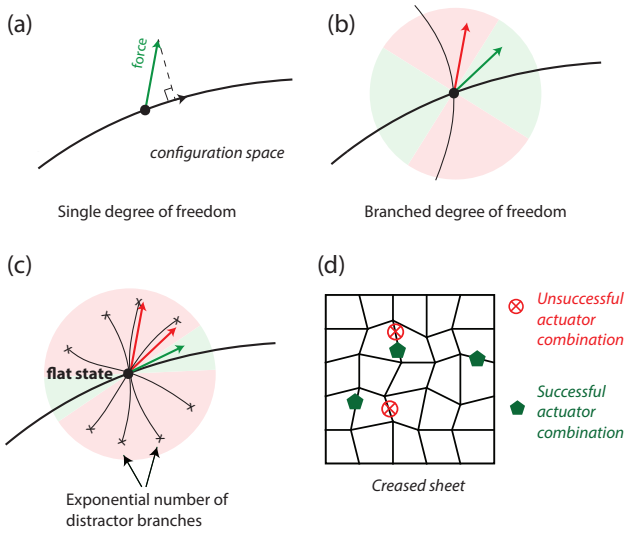

Figure 3.1: Bifurcated folding motions
(a) Structures designed with only one folding motion ('mechanisms') are thought to be easy to control since any applied force not exactly perpendicular to that motion will actuate it. (b) However, if a mechanism has a branched degree of freedom (bifurcation), the applied force (green) must make a smaller angle with the desired branch than with the undesired branch. (c) We show that programming a stiff sheet with one folding motion inevitably creates an exponential number of other dead-end 'distractor' branches that are of zero energy only to linear order. The applied force needs to be highly aligned with the desired folding motion in order to avoid the distractors. (d) Consequently, we must actuate multiple creases in a carefully selected combination (green) to successfully fold a self-folding crease pattern.

bles earlier works, such as Bern and Hayes' classic result on NP-hardness of flat-foldability [46] and others [34, 72, 73, 74]. However, Bern and Hayes focused on the ordering of folds in multi-stage folding, also investigated later in [48, 75, 76]. Here, we focus on self-folding sheets with a single temporal stage. More significantly, many earlier works [46, 74] concern the *computational* difficulty in finding a consistent global Mountain-Valley assignments (e.g., 'forcing sets' [72, 73]), while our work concerns whether the *physics* of folding can find a desired global Mountain-Valley assignment, taking into account physical effects such as mechanical advantage and energy landscapes that play no role in these earlier works. A recent work [77] considers similar actuation questions for single vertices and simple loops of vertices; in contrast, we use an energy model and focus on statistical results for large quadrilateral meshes with an exponential number of distractors.

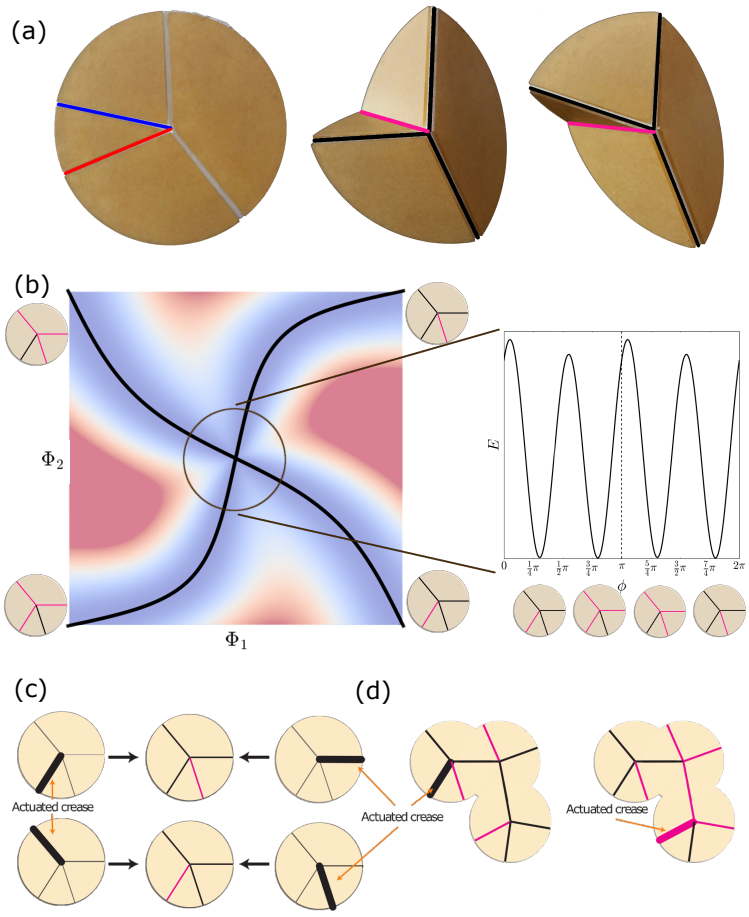

Figure 3.2: Bifurcations for vertices and chains of vertices
(a) A single vertex has two distinct folding branches, in which 3 creases form a mountain fold (black line) and 1 becomes a valley fold (magenta), or vice-versa. The distinct branches are identifiable by the odd-one out crease which folds opposite to the rest, with the two choices marked by blue and red lines. (b) The two branches meet at a bifurcation at the flat state. Φ_1 and Φ_2 correspond to the two eigenvectors of folding angles that span the 2d null space of the vertex. Given a folding amplitude (e.g. $\|\vec{\rho}\| = 0.1$), we blow out the energy of any configuration on a circle of that radius, with two clearly identified zero-energy configurations (and their negatives). (c) When a selected crease is actuated, the vertex chooses the branch in which that actuated crease folds more relative to other creases (rule of mechanical advantage). Since the odd-one-out crease and its transverse crease tend to fold less than the other crease pair, the odd-one-out crease is generally adjacent to the actuated crease. (d) When N vertices are linked together into an open-ended chain, the chain can fold in 2^N different folding branches. Given an actuated crease, the resulting MV data can be predicted by applying the branch selection rule of (c) to vertices in sequence, as each successive vertex is actuated through the crease linking it to the prior vertex.

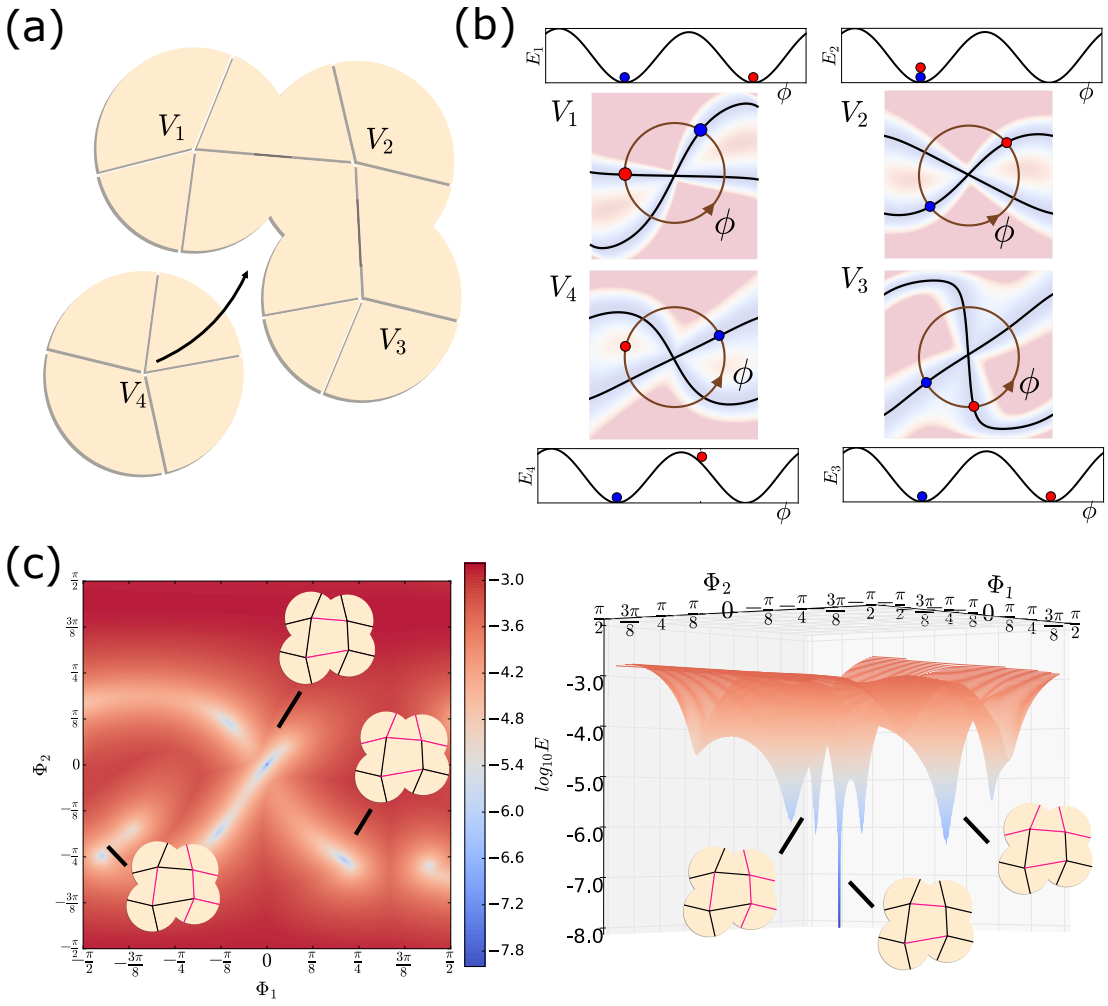

Figure 3.3: Loops of vertices give rise to a glassy landscape

(a-b) When a chain of vertices is closed by adding a final vertex, the resulting branches are no longer of zero energy. E.g., if vertices V_1, V_2, V_3 are at one of their two zero energy states (red dots), V_4's folding state is constrained since the folding state of creases $V_1 - V_4$ and $V_3 - V_4$ are already set. The resulting energy for V_4 is generically not zero (red dot for V_4). We computed the energy in the $4d$ linearized null-space at a fixed norm $\|\vec{\rho}\|$. (c) The heatmap and surface plot show a 2d projection onto the two top eigenvectors Φ_1, Φ_2 in the linearized null space.

3.1 Results

3.1.1 4-vertex and chains of 4-vertices

When a vertex with n creases is folded, the n dihedral fold angles $\rho_i, i = 1, \ldots, n$ are related by 3 equations [41]. Thus n-valent vertices with $n \leq 3$ will be completely rigid, while vertices with $n \geq 5$ have multiple degrees of freedom. 4-vertices are of special interest as they have precisely one degree of freedom.

However, a crucial caveat to this Maxwell counting is that only 2 of the 3 vertex equations are independent when the vertex is laid out flat [77] (i.e., unfolded). Consequently, it was shown [12] that a generic 4-vertex has two distinct folding branches that meet at a bifurcation at the flat state (Fig. 3.2). To see this quantitatively, we follow Tachi's use of rotation matrices [37, 42] to derive three constraint equations $T_a(\vec{\rho}; \vec{\theta}) = 0, a = 1, 2, 3$ associated with the vertex where $\vec{\rho}$ are the fold angles at creases and $\vec{\theta}$ are the in-plane angles between creases (see appendix B). We expand the constraints T_a in a series in ρ_i about the flat state $\vec{\rho} = 0$ as $T_a(\vec{\rho}) = C_a^i \rho_i + D_a^{ij} \rho_i \rho_j + \ldots$ (where repeated indices are summed over). Configurations that violate these constraints will have $T_a(\vec{\rho}) \neq 0$ and we can associate an energy $E_{Vertex} \equiv \sum_a T_a^2$ with these configurations. See Appendix B for more discussion on alternative choices of energy.

Such an energy of a general vertex configuration scales as $\|\vec{\rho}\|^2$. However, C_a^i has rank 2, giving a two dimensional space of zero modes in the linear approximation about the flat state $\vec{\rho} = 0$. The energy scales as $\|\vec{\rho}\|^4$ for folding modes in this linearized null space (with no bent faces). Fig. 3.2b shows the energy for folding modes within the linearized null space as we fold to larger angles. We see that two special folding branches within the linearized null space have zero energy to all orders. Thus, a generic 4-vertex has a full $2d$ vector space of zero modes at the flat state in a linear approximation, but only two $1d$ branches of zero energy upon non-zero folding. This is consistent with Maxwell counting, as one constraint is redundant at, but only at, the flat state.

The two folding branches differ qualitatively in the sign of their fold angles. Both branches

satisfy the following rule [78, 79]; three of the four creases must fold in a common orientation (say, 'Valley' fold) with the final 'odd-one-out' crease folding the other way ('Mountain' fold). The final odd-one-out crease can be either one of the two creases whose neighboring angles add to less than π; see Fig. 3.2a. This discrete choice gives rise to the two branches. Note that the two creases capable of being the odd-one-out are always adjacent.

3.1.2 Branch selection through mechanical advantage

When external folding torques $\tau_i, 1 = 1 \ldots 4$ are applied to the creases of a 4-vertex and released, the vertex will relax into one of the two branches (Fig. 3.2a) with corresponding folding angles $\vec{\rho}_\alpha, \alpha = 1, 2$. In the linear regime $\|\vec{\rho}\| \ll 1$, using our energy model (Eq. 3.1) we find that computing the normalized dot product between the applied vector of torques ('applied force') $\vec{\tau}$ and the folding angles $\vec{\rho}_\alpha$ of the two branches identifies the actuated branch; the vertex will relax into the branch with higher dot product $\tau \cdot \vec{\rho}_\alpha / \|\vec{\tau}\| \|\vec{\rho}_\alpha\|$. This rule is equivalent to selection based on mechanical advantage; when one crease is actuated, the vertex folds into the branch in which that crease's folding is larger relative to other creases (i.e., contributes more to the norm $\|\vec{\rho}_\alpha\|$).

Our mechanical advantage rule is based on a model energy landscape where the angular bisector of the two branches separates their attractor basins. In real material vertices, the dividing line between the attractors might be closer to one branch than the other (appendix C); such complications do not change our results qualitatively. In contrast, a recent work [77] assumed that actuation might fail if the applied force has a positive dot product with *any* other available branch. In such a model, even applied forces perfectly aligned with a branch may be classified as incapable of evoking that branch, in contrast to energy landscape-based models.

Our mechanical advantage rule can be restated as a heuristic in terms of Mountain-Valley (MV) choices. In either folding branch, the crease with odd-one-out MV state and its transverse crease fold less than the other pair of creases that share a common MV state [77] (To see this intuitively, consider the limiting case in which all in-plane angles are nearly 90 degrees and the vertex folds in half along one pair of creases with the same MV state; the other pair of creases barely fold at all).

Combining this observation with the dot product rule, we conclude that when a single crease is actuated, the vertex will choose the branch in which the crease transverse to the control crease will fold with the same MV state (Fig. 3.2c).

This branch-picking rule is easily extended to chains or trees of vertices, as long as no loops are present. If we actuate at one select crease at a vertex in this chain, we can determine the branch choice at that vertex using the above rule and thus the MV state of all creases at that vertex. Any neighboring vertex is actuated by the creases connecting them. In the absence of loops, there is only one path from the controlled vertex to any other and hence the mode-propagation rule unambiguously determines the branch choice at each vertex (Fig. 3.2d).

In this way, for any given actuated crease, the branch selection and propagation rule unambiguously selects one branch out of the 2^N bifurcated folding branches of an N vertex chain. Thus, at least with idealized materials, a choice of branch can easily be made in the loop-less case. In contrast, we will now show that patterns with loops, even if made from idealized materials, are intrinsically difficult to fold.

3.1.3 Loops of vertices create glassy energy landscapes

If 4-vertices are connected around a loop, we can no longer make an independent choice of folding branch at each of the vertices. For example, for a loop of four 4-vertices like that in Fig. 3.3a, we can make independent branch choices for three of the vertices - say for V_1, V_2 and V_3 - which puts them in one of their zero energy states (red or blue points in Fig. 3.3b). The final vertex's folding branch is then completely determined because the state of two creases at V_4 are already determined (namely, creases $V_3 - V_4$ and $V_4 - V_1$). Generically, the resulting state for V_4 will not be of zero energy [13] (red dots in Fig. 3.3b). We thus find that the resulting folding branch is of non-zero energy, unlike for chains of vertices.

Such thin sheet configurations with non-zero energy will also show face bending. We add a stiff face diagonal to each inner face in a crease pattern with face stiffness parameter κ_f (see Appendix B for more details, specifically on the balance of stretching and bending). The energy model of a

generic configuration with loops thus becomes

$$E \equiv E_{Vertex} + E_{Face} = \sum_a T_a^2 + \frac{1}{2}\kappa_f \vec{\rho}_f^2 \tag{3.1}$$

where $\vec{\rho}_f$ are the face bending angles.

Going through the $2^3 = 8$ independent branch choices for V_1, V_2, V_3 (which then determine the state of V_4), we should expect to generically find 8 branches of non-zero energy. In fact, these folding branches are of zero energy to quadratic order but of non-zero energy at next order; i.e., the energy of these branches scale as $\kappa\rho^4$ with $\kappa \neq 0$. In contrast, $\kappa = 0$ for all the 2^N folding branches of a chain of vertices. To gain more intuition about these branches and their energies, we fixed the overall folding magnitude $\|\vec{\rho}\|$ for a single 4-loop and computed the energy as a function of the angular directions in $\vec{\rho}$ space. A two-dimensional projection is shown in Fig. 3.3c where each branch shows up as a local minimum with depth proportional to κ.(See Appendix B for more details, including accounting for finite face bending and stretching energies at different $\|\vec{\rho}\|$).

Thus, we find that loops of vertices have a glassy folding energy landscape, much like a spin network with frustrated loops [80], and unlike trees or chains of spins.

A desired branch's energy can be made arbitrarily low or even zero to all orders in folding by fine-tuning in-plane angles using 'loop' equations [21, 43]. While the design process can make a desired folding branch be the ground state of the landscape, it does not change the glassy attractor structure shown in Fig. 3.3c; see appendix A (Fig. 3.6) for comparison. Different actuated creases initialize the folding process in different parts of the glassy landscape; folding then involves flowing downhill to a local minimum. Hence, actuating a desired branch in such a landscape can be difficult in the presence of a multitude of distractor branches.

Large patterns - number, attractor size of distractors

Large patterns made of many 4-vertices contain many loops and the number of distractor branches grows rapidly. We generated quadrilateral meshes of random geometry made of $\sqrt{A} \times \sqrt{A}$ vertices,

folded each mesh with random applied forces $\vec{\tau}$ and allowed it to relax into a local energy minimum until no more new minima were discovered. In this way, we determined the following landscape properties:

(a) The total number of distinct branches $N_{branches}$ for a given quadrilateral mesh grows exponentially with the size of the mesh, with the precise number of minima depending on the distance $||\vec{\rho}||$ from the flat state at which folding is stopped (Fig. 3.4a).

The increase in the exponential number of minima with folding distance arises because of the well-studied relationship between stretching and bending in thin sheets [54, 55, 21]. As discussed in Appendix B, close enough to the flat state, face bending and stretching energies are comparable. In this regime, as suggested by Fig 3a, choosing the states of three vertices around the loop strongly constraints the state of the fourth vertex. As one folds more, face bending becomes less expensive than stretching for thin sheets. Consequently, constraints on the fourth vertex weaken, revealing a larger (but still exponential) number of branches at larger $||\vec{\rho}||$, shown in Fig. 3.4a.

Note that while our numeric results unambiguously show exponential growth for all the $||\vec{\rho}||$ shown in Fig 4a, we find that the precise form cannot be numerically determined with confidence, despite $N_{branches}$ varying over 2.5 orders of magnitude. Note that in the limit of free face bending, our self-folding mesh is transformed into the fully triangulated patterns studied recently in [81].

(b) The attractor size of each distractor branch (taken to be the fraction of random actuation forces that actuate the branch) is generally small; see Fig. 3.4b. The largest attractor for the 4×4 mesh sampled is only $\approx$ 15% i.e., only 15% of random torques will actuate that branch. Most branches have far smaller attractor basins. The typical attractor size is expected to drop sharply with A. Data for small patterns suggest up to $A = 64$ suggests a power law dependence of the mean attractor size.

Actuation of large loopy patterns

How many creases need to be actuated - and which ones - to pick the desired branch in a landscape with an exponential number of other minima? Such landscapes arise in diverse areas of physics

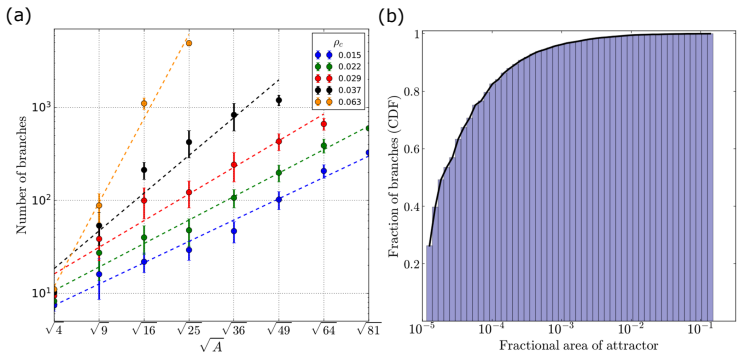

Figure 3.4: Large patterns have an exponential number of branches (i.e., minima) of decreasing attractor size
We characterized the landscape by sampling random quadrilateral meshes of size up to $A = 81$ vertices and folded each one with random torques until no new stable branches were found ($\kappa_f = 10^{-6}$, see Eq. 3.1). (a) Quadrilateral meshes show an exponential number of distinct folding branches (i.e., local minima) in its energy landscape; the precise scaling depends on the total extent of folding $\|\vec{\rho}\|$, reflecting the relative importance of bending and stretching energies (Appendix B). (b) The size of attractor basins around different branches for a fixed pattern does not exceed 15% of the total space for a 4×4 mesh.

for the same reason – frustrated disordered interactions – and are often referred to as 'complex' or 'glassy' [62, 67, 71, 80, 82].

To answer this question for self-folding origami, we study a random pattern with a chosen branch, shown in Fig. 3.5a. Since the crease locations at which folding torques are applied can be better controlled than the precise magnitude of torques in many applications [10], we applied folding torques of fixed magnitude to different randomly selected subsets of creases. The applied torques were always of the correct sign (mountain or valley) needed at that crease for the chosen branch. As seen in Fig. 3.5b, actuators are needed on 18 out of a total of 60 creases to have a 50% probability of folding the pattern.

For applications where the precise torque magnitudes can be controlled in addition to location (as explored recently in [77]), we must characterize how closely the applied vector of torques must align with the folding angles of the desired branch (see Fig. 3.1). We present such results on dot products in appendix D.

Requiring a large number of actuators or precise control of torque magnitudes defeats the

purpose of designing a single degree of freedom mechanism; it is hard to call a system requiring such delicate control 'self-folding'.

How then can self-folding origami be folded with a minimal number of actuators? A lesson can be drawn from similar glassy landscape search problems in models of protein folding (e.g., Levinthal's paradox [67, 69, 70, 83]) and related NP-hard satisfiability (SAT) problems [71, 84] that vary from the Traveling Salesman Problem to Sudoku [85]. A common element in these satisfiability problems is that random seeding of the search for the global minimum leads to repeated backtracking after reaching local minima, both in the context of computer algorithms (as the DPLL algorithm for k-SAT [71]) or for physical dynamics (as in protein folding) [84]. However, careful seeding of the search - e.g., if the right boxes are filled in first in Sudoku [85] or if the right parts of the protein are folded first - can greatly reduce or even eliminate backtracking [71] before reaching the global minimum.

Correct seeding is even more important for origami since folding is assumed to happen at 'zero temperature' (e.g., without any noise or fluctuations). As a result, the structure cannot backtrack out of a local minimum as in the case of non-zero temperature SAT problems [84].

3.1.4 Folding islands

To understand the role of frustration and seeding in the origami context, we must consider both the branch selection rule and the effect of loops. Even in the absence of loops, when an actuated vertex is folded into its desired branch, the MV state propagated to a target vertex as shown in Fig. 3.2d, can disagree with the desired folding branch at the target vertex and thus fold it incorrectly.

The situation is complicated by the presence of loops since a target vertex can be reached from a control crease by multiple paths. The mechanical advantage heuristic applied to different paths, which reach a particular vertex from different directions, may not be consistent. For example, different paths might imply different folding branches for the target vertex. Thus, in the presence of loops, the mechanical advantage heuristic (or any other such path-based heuristic) is not sufficient to folding all vertices into desired branches.

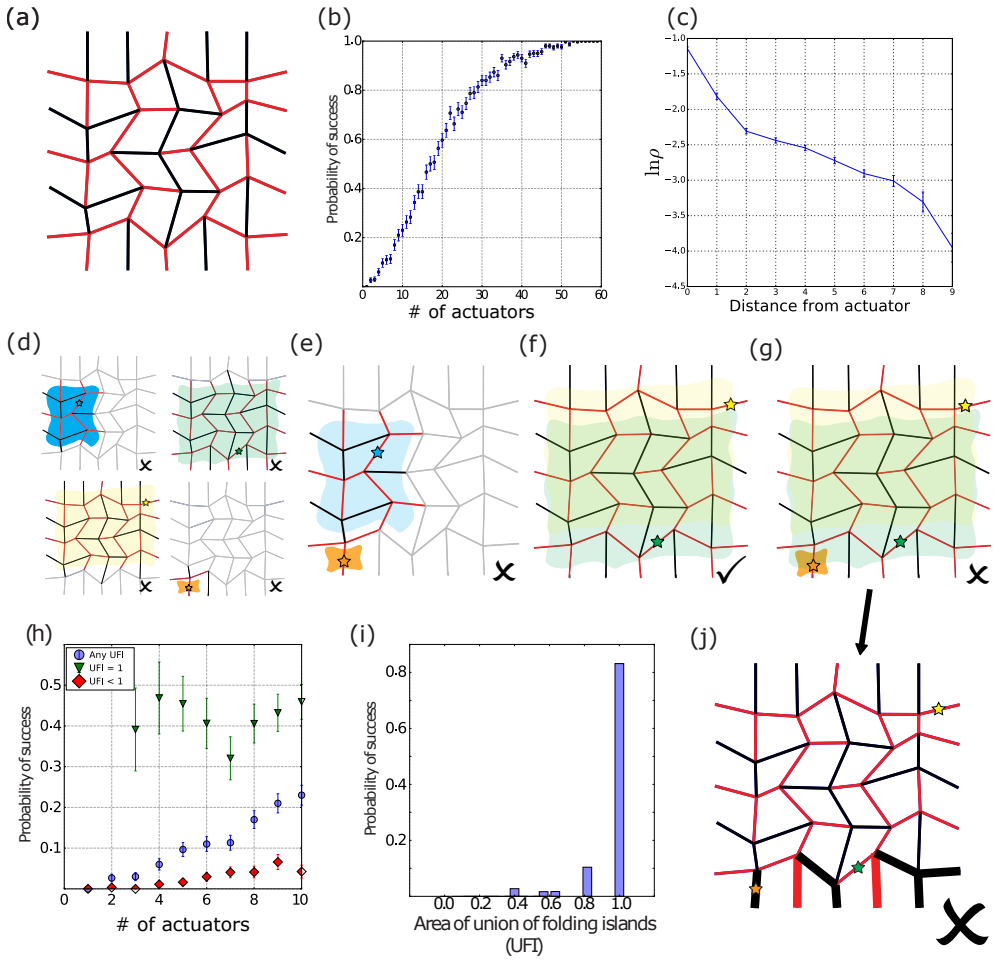

Figure 3.5: Spatial distribution of actuators determines folding success
(a) A 4x4 quadrilateral mesh pattern, with its designed soft branch indicated by line colors (black - mountain, red - valley). (b) If standard actuators are placed on randomly chosen creases of the pattern, at least 18 actuators ($\sim$ 30% of creases) are needed to have a 50% chance of successful folding. (c) If just one crease is pressed, the resulting branches typically have a decreasing folding magnitude for creases away from the actuated crease. (d) The 'folding island' of a crease is the largest sub-pattern that folds correctly when cut out from the full pattern and actuated at that crease. The area of the union of folding islands relative to the entire pattern (denoted UFI) provides a simple design heuristic; (e) Actuated crease sets with $UFI < 1$ generally do not successfully fold the pattern while (f) actuators with $UFI = 1$ are successful. (g) However, successful actuation can sometimes be ruined by actuating an additional crease (orange) with a small folding island. (h) Actuated crease sets of given size are dramatically more likely to fold successfully if their $UFI = 1$ (green) rather than $UFI < 1$ (red). (i) Using the data in (b) we find that UFI is a sharper predictor of success than the number of actuators. (j) Folding islands also explain a counter-intuitive effect where successful actuation can sometimes be ruined by actuating an additional crease (orange) with a small folding island. The obtained branch is different than the designed one primarily in creases close to the "bad" actuator (thick lines).

As a result, while a designed folding branch guarantees a *globally* consistent configuration of vertex branch choices (e.g., blue dots in Fig. 3.3b), such a global configuration may be difficult to reach using the *local* MV propagation rule in Fig. 3.2c from a single actuator. Hence successfully folding a large pattern in a desired branch can require actuating multiple creases at the same time.

A clue to finding good sets of actuators is seen in Fig. 3.5c: When actuated at a single crease, the obtained (undesired) branch is generally 'localized', where folding amplitudes near the actuated crease are larger than far away from it. This suggests that globally desired folding might be achieved with cooperative local actuators, each of which fold their local neighborhoods correctly.

To find the number of actuators needed for successful actuation of a desired branch, we identify unfrustrated sub-patterns called 'folding islands'. We define the folding island of a crease (with respect to a desired folding branch) as the largest contiguous region of the pattern that will fold in the desired branch, if that region is cut out and actuated at the chosen crease. For computation of the folding islands see appendix E. Fig. 3.5d shows that folding islands for different creases can vary greatly in size and generally do not cover the whole pattern. While folding islands can be approximately deduced using the simple MV propagation rule in Fig. 3.2c, the exact shape depends on the precise in-plane angles.

These considerations suggest a heuristic necessary condition for a set of actuated creases to fold a pattern; the union of their folding islands should cover the whole pattern. If not, as in Fig. 3.5e, when folding reaches the boundary of a folding island, folding will jam in a high energy distractor branch as vertices outside the union of islands will fold incorrectly. On the other hand, the two actuators shown in Fig. 3.5f, whose folding islands together cover the entire pattern, successfully fold the pattern.

Folding islands provide a new perspective on why randomly placed actuators (Fig. 3.5b) were poor at folding the pattern. In Fig. 3.5h, we went through the different actuated crease sets used in Fig. 3.5b and computed the area of the Union of Folding Islands (which we denote UFI, defined as the fraction of all creases belonging to the union) for each set. We see, for example, that a set of 5 actuators is 60× more likely to fold the pattern if it has $UFI = 1$ rather than $UFI < 1$. Similarly,

Fig. 3.5i shows all the data in Fig. 3.5b, but plotted against UFI instead of number of actuators. These results show the union of folding islands and thus spatial placement of actuators is a much better predictor of folding success than the number of actuators (We find a few cases of successful actuation e.g., at $UFI = 0.8$ when the folding islands cover most vertices). In particular, the condition $UFI = 1$ eliminates many spatial arrangements of actuators that are nearly guaranteed to fail.

Folding islands also shed light on a counter-intuitive phenomenon shown in Fig. 3.5g. While the two actuators in Fig. 3.5f can successfully fold the pattern, adding another actuator with a very small folding island as in Fig. 3.5g, can stop the previously successful folding! Fig. 3.5j shows the resulting undesired branch, which is different than the desired branch in the bold creases. Thus, the interaction between the green and orange actuators leads to misfolding mostly in proximity to the orange actuator (just outside of its small folding island), consistent with the falling influence of actuators with distance as shown in Fig. 3.5c. Such effects reduce the probability of success in Fig. 3.5g when $UFI = 1$ to be less than 1. Predicting these subtle competition between different control creases requires knowledge of the precise in-plane angles of the pattern and we are unable to formulate a strict necessary and sufficient condition for successful folding without full pattern information. Nevertheless, identifying the folding islands provides a useful design heuristic to greatly reduce the number of actuators needed, as seen in Fig. 3.5h,i. See Appendix E for more on how folding islands can be incorporated into an algorithmic framework for actuator placement that is vastly more feasible than blind search (experimental or in simulations) through all combinations of actuators.

3.2 Discussion

Sheets with crease patterns designed to exhibit exactly one folding behavior are nevertheless difficult to fold. We traced this difficulty to the fact that stabilizing one folding behavior using frustrated interactions between binary degrees of freedom (bifurcated origami vertices [12, 41])

inevitably stabilizes an exponential number of other distractor behaviors – i.e., a 'complex' or 'glassy' landscape [82]. Thus our results establish fundamental limits on the programmability of energy landscapes for sheets, paralleling similar limitations in other bottom-up approaches such as self-assembly of particles [61] and self-folding of polymers [62] as well as classic NP-hard satisfiability (SAT) problems [71, 84].

Self-folding with real materials can introduce other complications, specific to those realizations, that make folding *more* difficult than described in our paper. In appendix C, we explore several other models of self-folding sheets, incorporating stiffness of creases, variable bending vs stretching energy of sheets using COMSOL, and manufacturing error in crease placement. We find that the statistical properties of the glassy landscape remain unchanged. Thus, our work points at a fundamental glassy difficulty that is intrinsic to self-folding, reliant only on frustrated interactions between bifurcated 4-vertices – and hence must be faced by *any* material realization.

We saw that many actuators are needed to successfully fold self-folding sheets, if their locations are randomly chosen. However, carefully choosing the set of actuated creases can reduce their number dramatically. We interpreted successful combinations in terms of unfrustrated sub-patterns called folding islands that successfully fold when cut out of the full pattern. The connection to protein folding and other NP-hard problems drawn here suggests other ways forward, including temporal staging, folding funnels and chaperoned folding [62, 71].

Recent self-folding origami applications vary greatly in the materials used and in actuation mechanisms for active hinges, including electric [10], optical [56], thermal [49] and chemical (pH) [50] methods. In many applications, energy can be selectively input to specific creases, e.g., by controlling the electric current to shape-memory polymer hinges [47, 48] or light input to hydrogels [13]. Our work suggest which combinations of creases should be given energy input for successful folding, even showing how adding an actuator can ruin successful folding (Fig. 3.5h). Going beyond self-folding patterns, our considerations also apply to each temporal stage of multi-stage sequential folding patterns [48, 75, 76].

The folding difficulty described here and the resulting need for careful actuation mathemati-

cally applies only at the flat state; but since the energy barriers between branches grow more slowly with folding for a softer sheet, careful actuation needs to be maintained until a larger folding angle for soft sheets.

Recent experiments on controlled repeated crumpling and extension of sheets suggests an inability to refold along existing creases, leading to the formation of new creases [86]. While the 4-vertex patterns studied here are not good models of crumpled soft paper with significant face bending, our results do suggest that the difficulty of refolding a crease pattern, and thus the propensity to create new creases, grows with the softness of the sheet and when unfolded closer to the flat state.

3.3 Appendix A - Design of folding branches and the energy landscape

An origami pattern containing loops made of 4-vertices inevitably has many folding branches of finite energy. For a generic pattern of some particular topology (e.g. a 'quad' made of four 4-vertices in a loop), the folding branches have non-zero energies distributed over a wide range when folded with the same magnitude $\rho \equiv \|\vec{\rho}\|$ (Fig. 3.6a). However, all these energies scale as $E \sim \rho^4$, since the folding branches are within the null space of the linear part of the expression for energy.

For many applications, one requires softer folding branches, that could be folded to a non-linear extent (large ρ) with a small input energy. Designing such soft folding branches can be accomplished by fine-tuning in-plane angles using 'loop' equations [43, 21]. The process entails picking a branch and modifying the geometry of the pattern to make that branch gradually softer. Each successive loop equation, when solved exactly, changes the scaling of the folding energy for the designed branch

$$E(\vec{\rho}) \sim \rho^{2+4n}, \tag{3.2}$$

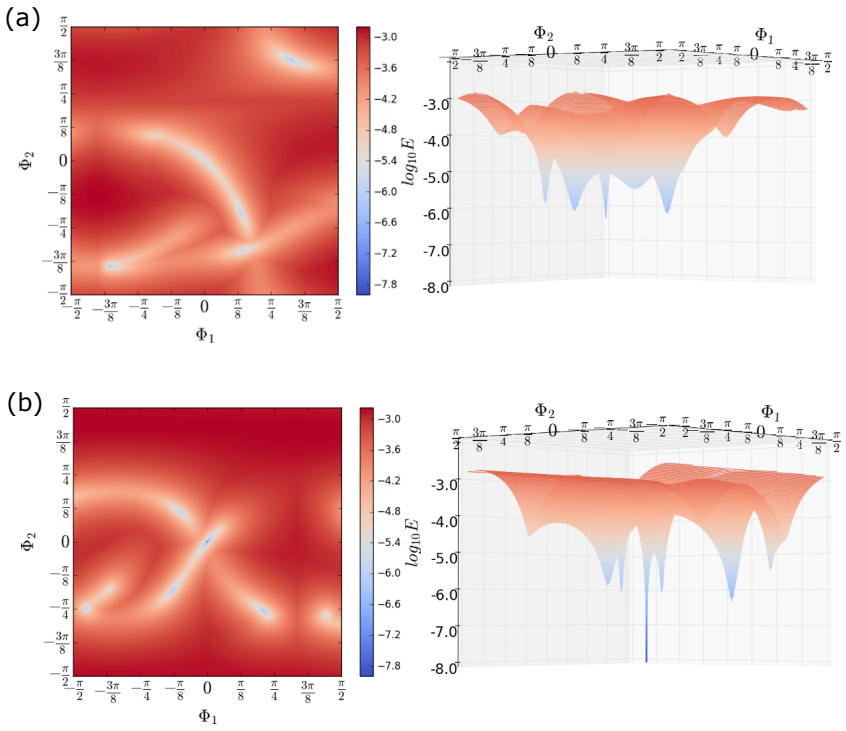

Figure 3.6: Energy landscape of a quad loop pattern at fixed norm of folding angles $||\rho||$ (a) A generic quad loop pattern has $3-7$ folding branches seen as minima in the energy landscape (Left - heat map; Right - corresponding 3D surface). Generic minima have energies that scale as $E \sim \rho^4$. (b) By solving loop equations [21, 43], we can design one specific folding branch to be qualitatively softer than all others (identified as the much deeper minimum). The rest of the branches retain their original energy scaling and thus become distractors in the energy landscape.

where n loop equations are solved in sequence.

We can thus design a specific branch of the pattern to be a soft as needed, but what becomes of all the other folding branches of the pattern? We find that the other branches, for which no loop equations are solved, remain qualitatively the same. Their number in general does not change, and neither does their energy. As their energy still scales with ρ^4, we are left with a distinct energy landscape, where the one designed branch is qualitatively softer than all of the rest (Fig. 3.6b). We thus call the other branches 'distractors', as they correspond to high energy minima in a glassy landscape, with energy much higher than the ground state (i.e. the designed branch).

A remarkable fact about the distractor branches is that they do not have to comply with single vertex rules. One might find a distractor branch in which a few vertices have 'illegal' (non-

Kawasaki-Justin) configurations, e.g. 4 valleys/mountain, 2 valleys/mountains, etc. Clearly, vertices with such configurations must contribute to the energy of the entire pattern, but they can do so within the linearized null space of the pattern. It is notable that all designed soft branches must be 'legal' at the single vertex level. The energy of any configuration containing 'illegal' assignments for any vertex scales at least as $E \sim \rho^4$.

3.4 Appendix B - Energy and vertex constraints

A single 4-vertex has 2 zero energy folding branches that extend to arbitrary overall folding magnitude $\|\vec{\rho}\|$. These modes cost no energy as the corresponding choices of $\vec{\rho}$ exactly satisfy the vertex constraints.

Vertex constraints are derived for any closed vertex by considering a condition such that the vertex does not tear open when folded [37, 42]. A small disk surrounding the vertex configuration is defined by the folding angles ρ_i and the in-plane angles between the creases θ_i. One can 'walk around' the edge of the disk around the vertex and eventually return to to same point only if the vertex is not torn anywhere. This motion around the vertex consists of rotating with the θ_i angle about the central axis of the vertex, and then rotating about the dihedral angle ρ_i until one returns to the original position. If one orients the current vertex face such that the face occupies the xy-plane and the crease is on the x-axis, these two rotations can be expressed as [37, 42]

$$R_i = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\rho_i & -\sin\rho_i \\ 0 & \sin\rho_i & \cos\rho_i \end{pmatrix} \begin{pmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{3.3}$$

In general $R_i = A_i B_i$ where A_i is a rotation matrix about an axis along crease i by angle ρ_i while B_i is a rotation matrix about an axis perpendicular to face i by angle θ_i.

The condition that the vertex is not torn becomes

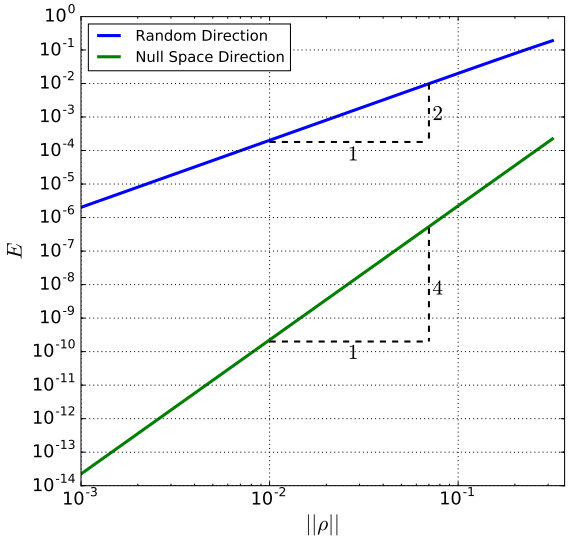

Figure 3.7: Stretching energy scaling in a 4-vertex $E(\|\vec{\rho}\|)$ depends on the direction of $\vec{\rho}$ For random directions we have $E \sim \rho^2$, while for directions within the linearized null space of the system $E \sim \rho^4$. Only two special directions (not shown) exist for which $E = 0$ to all orders in folding.

$$\prod_i R_i = I, \tag{3.4}$$

where the product is taken over all creases and faces i, and I the 3×3 identity matrix. Equation (3.4) can be shown to be equivalent to 3 independent equations for the off-diagonal upper matrix. Crucially, these are 3 non-linear constrains relating the folding values ρ_i around the vertex. A 4-vertex is a 1 DoF object, as 3 equation relate its 4 folding angles.

Near the flat state, all $\rho_i \approx 0$, and the matrices of equations (3.3-3.4) become essentially $2d$ rotation matrices about an axis perpendicular to the flat vertex; hence one vertex constraint is lost. The preceding considerations apply to all vertices constructing the pattern.

As shown in the main text, the constraints can be expanded about any configuration, in particular the flat state $\vec{\rho} = 0$:

$$T_a(\vec{\rho}) = C_a^i \rho_i + D_a^{ij} \rho_i \rho_j + \dots \tag{3.5}$$

Violation of these vertex constraints can be interpreted as a stretching energy at that vertex,

$E_{Vertex} = T_a^2$. Fig. 3.7 illustrates the vertex energy in this model. Random configurations of $\vec{\rho}$ for which $C_{ai}\rho^i \neq 0$ are characterized by an energy that grows quadratically in ρ. Configuration existing in the $2d$ linearized null space of the vertex have a scaling $E \sim \rho^4$. Large patterns made of 4-vertices have a high dimensional linearized null space where the energy scales like that of a vertex. However, we show in the main text that patterns with loops are frustrated, such that in general no zero energy folding branches exist.

In addition to vertex energy, when vertices form loops, one also expects face bending in thin sheets [54, 87]. We model such face bending by adding stiff face diagonals to each inner face; that is, we add face diagonals with torsional springs on them of stiffness κ_f and rest angle $\rho_f = 0$. Putting these together, our energy model is,

$$\begin{aligned} E(\vec{\rho}) &\equiv E_{Vertex} + E_{Face} \\ &= \sum_a T_a^2 + \frac{1}{2}\kappa_f \sum_{i \in faces} \rho_i^2, \end{aligned} \tag{3.6}$$

with κ_f a proper dimensional face stiffness factor. As a sheet is made thinner, the bending modulus is reduced relative to stretching [54, 55], effectively reducing κ_f.

Every folding branch of generic looped patterns balances stretching (i.e., vertex) and bending (i.e., face) energies, with relative importance depending on the folding amplitude ρ. To illustrate the importance of the different ρ scaling of the two energy terms, we counted the number of branches for quadrilateral meshes of different sizes, at different values of folding magnitude ρ_c (Fig. 3.8).

Note that the stretching term scales as ρ^4 (for branches in the linearized null space), while the bending term scales as ρ^2. Thus, at smaller folding magnitudes bending of faces is expensive and suppressed. In contrast, at large folding magnitudes stretching is more expensive and faces can bend much more freely. As explored in earlier work [21], the number of branches of origami patterns scales exponentially with the ratio of face bending to crease folding, consistent with these results.

Finally, while these models quantify the violation of constraints (using vertex energy and face

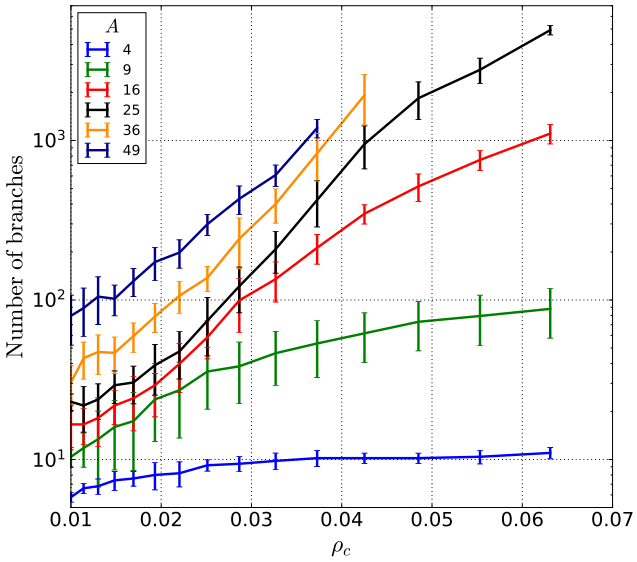

Figure 3.8: Number of folding branches for patterns of different sizes, folded to magnitude $\|\vec{\rho}_c\|$ At large folding angles the same patterns have many more folding branches due to the lesser importance of face bending compared to stretching. Main text Fig. 3.4a shows vertical slices of the same data, illustrating the exponential size scaling of the branch number ($\kappa_f = 10^{-6}$).

bending respectively) and a glassy landscapes emerges in this context, the results hold for arbitrarily stiff sheets. However, these results cannot be derived directly for strictly infinitely stiff sheets, a mathematical idealization that assign infinite energy to all non-zero-energy folding configurations. Consequently, that mathematical limit misses the glassy landscape that exists for arbitrarily stiff sheets. In this sense, softness is a singular perturbation and strictly infinitely stiff sheets are not a good approximation of realistic sheets, no matter how stiff, for the question of actuation.

Folding method

When external folding torques are applied to creases of a pattern, we fold the pattern by accounting for both the external and the internal forces generated by the energy model above (Eq. 3.1).

Given an external folding torque τ_i applied to crease i, the folding of an 'overdamped' but unconstrained crease follows $\gamma\dot{\rho}_i = \tau_i$, with γ the stiffness of the crease. In self-folding patterns, folding motions are $1d$ such that there exists preferable configurations ρ_i. This notion is encoded

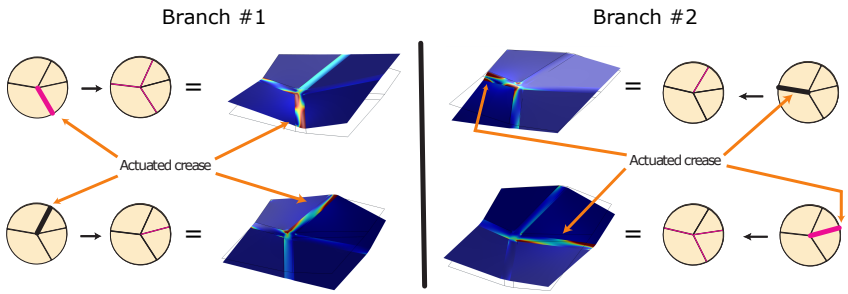

Figure 3.9: Finite element models of realistic sheets, simulated with COMSOL Multiphysics show that the mechanical advantage rule identifies the right folding branch. When a torque is applied to just one crease, the resulting branch is predicted by the mechanical advantage rule (Fig. 3.2c). We used the COMSOL shell model to simulate folding a realistic origami vertex (experiencing elastic strains). The folded configurations match those expected from our simplified vertex model defined only by vertex constraints, even though the COMSOL model accounts for many real world complications not accounted for by our simple model (e.g., finite crease stiffness and thickness, delocalized bending and stretching). Pattern parameters: Length $\sim 0.2m$, crease width $0.015m$, thickness $10^{-4}m$. Material parameters of the faces are density $\rho = 1760kg/m^3$, Young's modulus $Y = 8 \cdot 10^8 Pa$. Material parameters of the creases are density $\rho = 930kg/m^3$, Young's modulus $Y = 5 \cdot 10^6 Pa$.

by the energy model presented above, as unfavorable configurations require more bending of the faces. Thus, if left on its own, the pattern will change its configuration introducing internal torques $\gamma\dot{\rho}_i = -\frac{\partial E(\vec{\rho})}{\partial \rho_i}$.

These considerations allow construction of a folding algorithm, implemented as an ODE system:

$$\gamma \frac{d\rho_i}{dt} = \tau_i - \frac{\partial E(\vec{\rho})}{\partial \rho_i} \tag{3.7}$$

In practical computation we usually start the solution from the flat state $\rho_i = 0$ and solve this system with a fixed external torque τ_i until one of the dihedral angles reaches a certain predefined value (say, $0.5rad$). The equations are solved using MATLAB ODE solver.

3.5 Appendix C - Effects of material properties and imperfections

So far, we have considered a simplified model for self-folding origami made of stiff elastic materials. Bending energy is represented by stiff face diagonals, while all stretching energies are focused at the vertices as constraints. Creases themselves are free folding with no stiffness. Our model is a good approximation for self-folding origami in which the sheet is very thin and faces don't bend too much.

However, real origami made of realistic elastic materials is more complicated. This appendix illustrates how changing the energy model to incorporate the imperfections and complications of real materials - e.g., manufacturing error, crease stiffness, non-localized bending and stretching, finite thickness of sheets - modifies the configuration energy landscape.

The main results are that while these complications can indeed change details of the landscape, such as the precise number and energy of the minima, the underlying exponential structure and statistical properties of the landscape remain unchanged since they arise from the bifurcation of the 4-vertex. We thus conclude that while folding in *specific* real materials can face additional complications than those described here, our results point at foundational hurdles intrinsic to the concept of self-folding that must be faced by *any* material.

Stiff creases

So far we modeled idealized origami patterns in which the faces are stiff and the creases fold freely without any resistance. In real materials, the creases can offer some resistance to folding as well. We can model such complications by introducing a torsional spring on each crease. The energy model will be modified to:

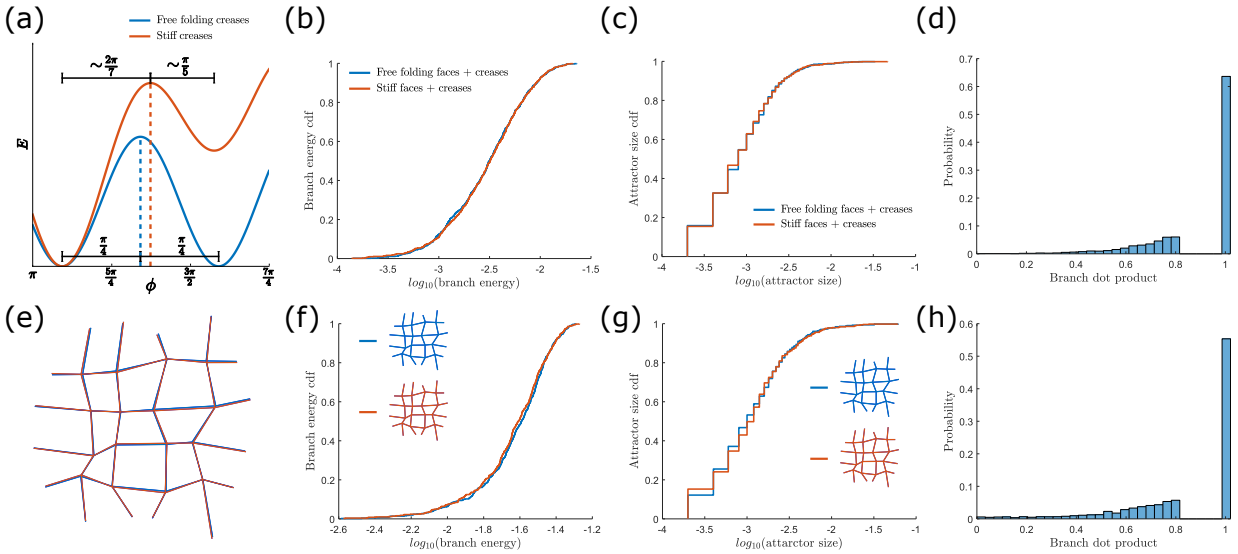

Figure 3.10: Varying material models changes landscape details but maintains underlying structure of exponential minima

(a) Simulating a 4-vertex with stiff creases results in the same two modes of an idealized vertex, as long as creases are not overwhelmingly stiff. The shift in minima position causes a moderate deviation from the mechanical advantage rule for sufficiently stiff creases. (b,c) Comparing a pattern with free folding faces and creases to the same pattern with stiff faces and creases, the number of branches remains similar. Furthermore, the distribution of branch energies and attractor sizes is statistically indistinguishable, implying the glassy structure retains the same statistics. (d) When branches of the patterns with free folding and stiff faces\creases are paired up, we find that most random attempts to fold the system enter the same branch for both patterns, showing how corresponding branches remain "close" in configuration space. (e) Realistic origami structures are expected to have manufacturing errors in vertex placement. The blue and orange pattern's vertices differ in location by 2% of the mean lattice spacing. (f,g,h) The same metrics as in (b,c,d) show how the landscape of patterns with manufacturing errors retain the same statistics as the originals.

$$E = E_{Vertex} + E_{Face} + E_{Crease}$$
$$= \sum_a T_a^2 + \frac{1}{2}\kappa_f \sum_{faces} \rho_f^2 + \frac{1}{2}\kappa_c \sum_{creases} \rho^2. \tag{3.8}$$

We compared the branching statistics of a 3x3 looped disordered mesh (Fig. 3.10e, blue) by considering $5 \cdot 10^3$ initial random torque schemes in the simplified and stiff crease models. Compared to the supplied external torque, the stiffness of faces was assigned a value $k_f = 0.1\tau$ and the stiffness of creases was $k_c = 0.02\tau$. The number of branches found for these two models were different by less than 1%. As shown in Fig. 3.10b,c the statistics of the landscape are also very similar; the energy of minima and their attractor region sizes have nearly identical cumulative distribution functions, showing that the glassy attractor structure is essentially the same. Another metric by which we can estimate how similar the landscapes are is by pairing up the closest attractors (in the dot product sense) between the two landscapes. We find that the majority ($\sim$ 64%) of the folding torque schemes lead to paired up branches in the two models. In contrast, only rarely does the same torque leads to approximately orthogonal branches in both models, showing that the expected resulting branches are well correlated.

Mechanical advantage rule

In the idealized model considered in the paper, we find that the mechanical advantage rule predicts the branching of a 4-vertex for any applied torque. The simplified 4-vertex will always fold into the branch that has the larger dot product with the applied torque. A graphical representation of this fact is seen in Fig. 3.2c, in which the ridge in the angular energy function is always located halfway between the two minima.

This idealization will be modified for more complicated (and realistic) elastic models, for which the energy ridge might be shifted closer to one of the branching minima. One way to model such alterations is again considering stiff creases (modified with torsional springs). We find that

the energy ridge can be moved towards one of the minima by increasing the stiffness of creases (Fig. 3.10a). However, a significant deviation requires very stiff creases that overwhelm the vertex constraints (meaning the origami pattern will more effectively bend and stretch than fold). As long as the pattern retains its folding topology, when creases are not too stiff and can be practically folded, the mechanical advantage rule is approximately correct.

To test this prediction under more real world complications, we simulated an origami vertex made of realistic materials with COMSOL Multiphysics (Fig. 3.9). This COMSOL model includes finite thickness of the sheet, creases of finite width, thickness and stiffness, face bending and stretching that is distributed generally over the sheet, among other complications that were not accounted for in our simplified model. Despite this, the mechanical advantage rule does in fact predict the correct branch folded by the tested initial torque schemes.

Finally, we emphasize that real world deviations from the mechanical advantage rule can only change the relative attractor size of the two branches at each vertex. The qualitative glassy landscape relies only on the existence of two branches at each vertex and not tied to their precise size.

Manufacturing errors

An additional complication to realistic origami patterns is the impossibility of perfectly manufacturing a designed pattern. Any physical manufacturing process will inevitably introduce errors in the pattern by placing vertices slightly off of their designed position. The effect of such errors on the energy of the designed branch was discussed previously [21], but even if the pattern could still fold into the designed branch, it remains possible that the designed actuation scheme will not work. This might happen if the designed applied torques (e.g. chosen by considering the folding islands method) now lead to a different uncorrelated high energy minimum (distractor) in the landscape.

We simulated two patterns, one with vertices displaced by 2% compared to the other (Fig. 3.10e). To check whether the energy landscape changed considerably due to this perturbation, we computed the same metrics as discussed previously in this section (Fig. 3.10fg) for $5 \cdot 10^3$ random torque schemes. The statistics of the energy landscape are once more essentially the same, with

the same distribution of branch energies and attractor sizes. Moreover, paired up branches between the two patterns (by dot product proximity) have highly overlapping attractors, as $\sim$ 55% of the random folding torques fold into paired up branches (Fig. 3.10h). However, there is indeed a significant probability that a designed actuation scheme for the original pattern will fail for the displaced pattern (in this sample $\sim$ 45%). Still, by controlling the manufacturing error one could make the energy landscape of the displaced pattern a better approximation of the original landscape, such that the success of actuation would be nearly guaranteed.

3.6 Appendix D - Dot product and attractor size

In the main paper, we mostly considered actuators applying equal torques to reflect many applications where the precise locations of actuators is easily controlled but the precise magnitudes of applied torque is not.

Here, we study vectors of folding torques $\vec{\tau}$ whose components might be variable in magnitude across the pattern. We might expect folding to be successful if the dot product $D = \vec{\tau} \cdot \vec{\rho}_{desired}/||\vec{\tau}||||\vec{\rho}_{desired}||$ between the applied torque and the desired mode is higher than the dot products $\vec{\tau} \cdot \vec{\rho}_{\alpha}/||\vec{\tau}||||\vec{\rho}_{\alpha}||$ with all distractor modes α. In practice, how large does D need to be for successful folding of the desired branch?

To determine the dot product needed, we actuated folding using random torques as in the main paper but now characterized success of folding as a function of the dot product (Fig. 3.11). We see that, for sufficiently large patterns, a dot product of $D \sim 1$ between the applied force and the desired branch is needed to have a significant chance of success.

Naively, a high dot product might seem easy to achieve; but note that in high dimensions (e.g. 4x4 patterns have a $60d$ configuration space), a vanishingly small fraction of all vectors (e.g., $\vec{\tau}$) have a non-negligible dot product D with any fixed vector (e.g., $\vec{\rho}_{desired}$).

In conclusion, large patterns require a high dot product $D \sim \vec{\tau} \cdot \vec{\rho}_{desired}$ between the applied vector of torques $\vec{\tau}$ and the folding angles of the desired branch $\vec{\rho}_{desired}$ to fold successfully;

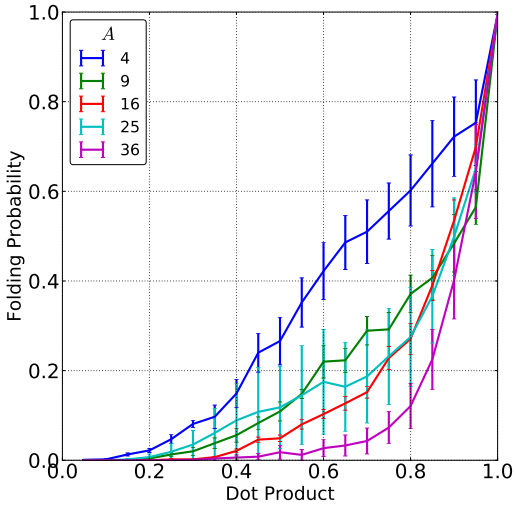

Figure 3.11: Large meshes require the applied vector of torques $\vec{\tau}$ to be closely aligned with the folding angles $\vec{\rho}_{desired}$ of the desired branch for successful folding
By folding random quadrilateral meshes of different sizes with random $\vec{\tau}$ and determining the success of folding, we find that the dot product $D = \vec{\tau} \cdot \vec{\rho}_{desired} / (\|\vec{\tau}\| \cdot \|\vec{\rho}_{desired}\|)$ in a large quadrilateral mesh must be close to 1. Since 'most' vectors in high dimensional space are orthogonal, only a vanishingly small fraction of applied vectors of torques can successful fold the desired branch.

since random vectors have vanishing dot products in high dimensional spaces, high dot product D is increasingly difficult to achieve. Such finely-tuned torques applied to every crease of a large pattern defeats the purpose of building 'self-folding' origami structures.

3.7 Appendix E - Computation of folding islands

As discussed in the main text, identifying the folding islands of the pattern provides a design principle for actuation schemes that successfully fold the desired branch. We find that choosing actuators whose union of folding islands covers the entire pattern dramatically improves the probability of successful folding. In this section we outline how folding islands are found, and some limitations regarding these procedures.

The folding island of a crease is defined as the largest contiguous region that can be folded successfully when cut out of the pattern. In principle, there might exist distinct folding islands of

equal area for a given actuator; in this case, we associate both of these maximal folding islands to the actuator with the understanding that when combined with another actuator, the more favorable folding island can be used. However, we did not find any such instances in our numerical exploration of patterns.

The method we employ to compute the folding island of a given crease is directly derived from the definition. (a) For the given pattern (e.g. Fig. 3.5a) and a given actuator crease, we first try folding a sub-pattern composed of just the two vertices connected to it. (b) We check whether any vertex in this set folds into the desired branch. If so, such vertices are included into the folding island. (c) We then enumerate the candidate vertices for the folding island found at the boundary of the current folding island - i.e., we enumerate all vertices connected to vertices already in the folding island. We pick a random member from this list, add it to the folding island and attempt to fold. (d) If the putative extended folding island folds correctly (including the new added vertex), the new vertex is included into the folding island. (e) We go back to step (c) and repeat until we can no longer add any vertices that fold successfully when the initial crease is actuated. When the process terminates, we are guaranteed a correctly folding region that is not contained in any larger folding region. (f) We repeat the entire process multiple times from scratch to explore alternative orderings of growth. The largest resulting folding islands over many runs is taken to be the true folding island.

Given enough trials, the algorithm will pick out the largest folding subset (defined to be the true folding island). With finite running time, the algorithm can, in principle, underestimate the size of the true largest folding island. In such a case, the heuristic of unions of folding island (UFI) is an even better metric than implied by Fig. 3.5 since some actuation schemes that worked at UFI < 1 should actually be described by UFI $= 1$. In practice, we found that testing putative boundary vertices by order of their distance (instead of a random order) from the actuator crease was particularly effective - this method quickly provided folding islands at least as large as those obtained after a small fixed number of random order trials, but is much faster. We used this faster method to compute the large number of folding islands needed for Fig. 3.5.

Design principle for actuator placement

Our results suggest a design principle for actuator placement: we first work out the folding island for each crease (through simulation or experiments). Then, using algorithms for the Set Covering Problem [88], we can identify minimal combinations of actuator creases whose folding islands cover the whole pattern. (The Set Covering Problem is a classic problem in computer science; given a set of subsets S of a 'universe' set U, one is asked to find a minimal combination of subsets S whose union is U. Many algorithms exist [88]. Here, U represents our vertices and S represents the set of folding islands of each vertex in the pattern.) Thus we have replaced a mechanically difficult problem at the time of actuation with a computationally difficult problem at the time of design. Further, by reducing properties of actuators combinations to a property of single actuators (i.e., their folding island), the above proposal is vastly more feasible than an exhaustive computational or experimental search through all combinations of actuators.

We can offer a rule of thumb for identifying which creases are likely to have large folding islands. Each vertex may be considered to "point" to two of its neighbors, along the transverse line whose two creases have the same sign of folding in the desired state. If the folding propagates from an actuator to the vertex via one of these neighbors, the vertex will fold correctly, by the mechanical advantage rule discussed above. Paths of pointing can be constructed, and if a crease is pointed to by many, long paths, it will tend to have a large folding island. The effect of loops and material complications means that this rule gives only a guide, not an exact prescription.

Applying these ideas to common patterns, note that Miura Ori has a sequence of long narrow folding islands along the zig-zag creases that fold in the same MV state. We previously showed [21] that origami patterns can be classified into three classes: Natural, Semi-natural and Unnatural. Natural patterns are the easiest to design and resemble Miura-Ori in that they contain one direction along which rows of creases are either all mountains or valleys; columns of creases in the perpendicular direction have alternating MV states. Such patterns should generally be expected to have long thin islands along the homogeneous direction. In contrast, semi-natural and unnatural patterns (e.g., that in Fig 3.5) tend to have compact islands.

Chapter 4

Shaping the topology of folding pathways in mechanical systems

When a heterogeneous mechanical structure like an elastic network or a thin sheet with creases is strained to large extents, it typically shows multiple stable states [12, 89]. As we vary the strain level, these states can smoothly deform and appear or disappear in bifurcations, creating a complex network of pathways in configuration space. The geometry and topology of such pathways determines which configurations are smoothly accessible from a given part of configuration space and which ones are not. The response of the material to applied forces is strongly shaped by the network of such pathways [77, 90, 91].

Such non-linear features of configuration space have proven to be a double-edged sword. When designed, multiple pathways and multistability can be exploited to create mechanical switches, shape-able sheets, and many other metamaterials [10, 12, 17, 26, 27, 60, 92, 93]. However, such non-linear features can also create problems [14, 17, 22, 81]. For example, self-folding origami, despite the name, has an exponential number of misfolding pathways that meet at a 'branch point' at the flat state [14, 31, 34, 36, 51], making it nearly impossible to fold into the desired folding mode [22, 81, 94, 95]. Similar 'branch points' in mechanical linkages pose challenges in robotics

and other applications [96, 97, 98].

In this work we suggest a design principle that sculpts the topology of dynamical pathways to desired and undesired states. We focus on elastic networks and creased sheets where rods or plates are connected at flexible joints. We show that heterogeneous stiffness in such joints can completely change the topological connectivity of non-linear pathways in configuration space. With a distribution of stiffnesses predicted by our equations, undesired pathways can be arranged to end in saddle-node bifurcations. Such bifurcations make undesired states inaccessible from parts of configuration space, at least in the limit of adiabatic folding. Finally, we find that pathways are accessible only at specific folding speeds, allowing dynamical selection between distinct behaviors.

While similar design principles to eliminate dynamical pathways to undesired states are common place in protein folding and self-assembly of macromolecular structures and viruses [65, 99, 100, 101], such ideas have not been systematically explored in meta-materials design.

Designing the topology of the bifurcation diagram presents several benefits. Once this topology has been designed for a material, it is not modified by entire classes of applied folding forces but determines the response to such forces. For example, in the context of self-folding origami, other approaches [77] have attempted to find fine-tuned folding forces that will fold a creased sheet successfully. In contrast, our approach produces systems that are truly 'self-folding' [11], i.e., our stiffened sheets fold along the desired pathway for almost arbitrary applied forces. Similarly, other approaches [10] have sought to introduce directional asymmetry so that, e.g., individual creases will fold in one way (say, Mountain) but not the other (Valley). Counter-intuitively, our approach shows that even symmetric stiffness in individual creases - an inevitable feature of real materials - can effectively pick a global Mountain-Valley pattern through their collective behavior.

We begin by showing that heterogeneous stiffness in hinges of a mechanical linkage changes the topology of folding pathways by creating saddle-node bifurcations. We show that hinge stiffness predicted from a Linear (or Quadratic) Programming problem can eliminate exponentially many undesired pathways at saddle-node bifurcations and demonstrate such an elimination for folding pathways present at the flat state of thin creased sheets. We show that such a stiffened thin

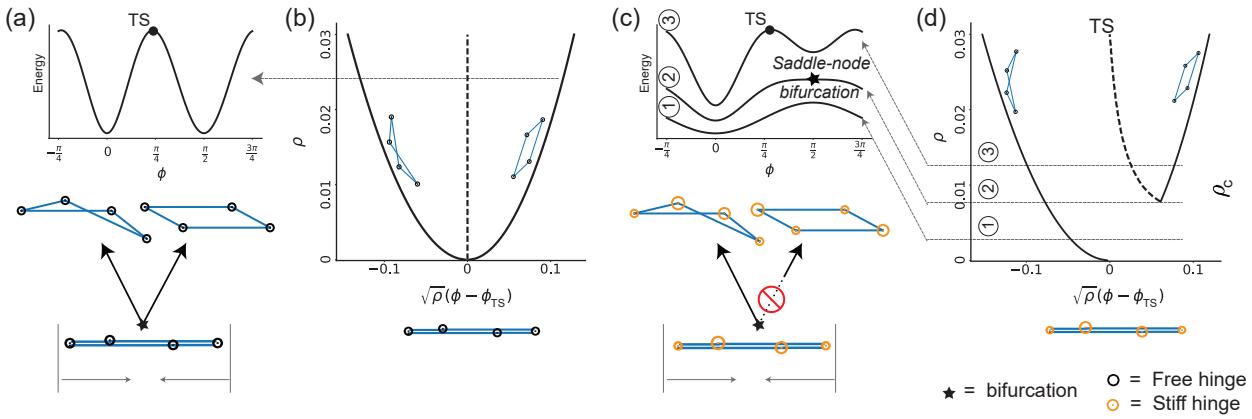

Figure 4.1: Stiff joints in a linkage network can change the connectivity of non-linear modes in state space
(a-b) The 4-bar linkage has only one degree of freedom but two distinct zero energy motions that meet at a branch point at the flat state, making the mechanism difficult to control. The two motions can be seen as minima in the energy landscape at a fixed total strain ρ. (c-d) We can eliminate a chosen motion in a saddle-node bifurcation at ρ_c by making the joints stiff to different extents (i.e., adding torsional springs that are relaxed in the flat state, larger orange circles denote stiffer springs). The bifurcation diagram shows that such a stiffness profile changes the connectivity of the two non-linear modes. One of the two modes is destroyed in a saddle-node bifurcation at ρ_c and is thus inaccessible from the flat state $\rho = 0$ (ϕ - angle variable in the 2-dimensional linearized null space at the flat state, see Supplementary Note 1).

sheet is truly 'self-folding' since the sheet can be folded robustly by a host of folding protocols and forces without any fine-tuning. Finally, we show that controlling the position of saddle-node bifurcations in configuration space, specific folding pathways can be made accessible at specific folding speeds. Consequently, we find that folding speed can select between different target structures.

4.1 Results

4.1.1 Avoided bifurcation in linkage networks

We first demonstrate our ideas on a simple but canonical model, namely the 4-bar mechanical linkage [102, 103] in Figure 4.1(a-b). While the structure has only one Maxwell degree of freedom, the flat state is a special point - it sits at a bifurcation where the degree of freedom is branched (and associated with a self-stress mode) [104]. When compressed as shown, the linkage must choose

one of the two distinct zero energy motions that conserve rod lengths. The associated energy landscape, at some fixed compression, has two minima corresponding to these motions with an energy barrier (transition state TS) between them; see Figure 4.1b. (See Supplementary note 1 for precise energy model.) Many studies [90, 91, 98] have sought to predict and eliminate such 'branch points' in complex mechanisms.

We take a different approach and observe that experimental realizations of such mechanisms [21, 105, 106] have imperfections that lift the energies of all the modes. If an imperfection can raise the energy of the undesired zero mode more than it raises the energies of the desired mode and the transition state TS, the undesired mode would disappear in a saddle-node bifurcation with the transitions state.

One such imperfection is stiffness in the joints. We model the stiffness of joint i by a torsional spring of stiffness κ_i that is relaxed in the flat configuration shown, i.e., at the branch point. That is, we assume a joint energy $E_i = \kappa_i \rho_i^2/2$, ρ_i being the folding angle measured from the flat state configuration.

We find that if the joints have unequal stiffness κ_i, the energies of different modes are lifted to different extents. In fact, one of the modes undergoes a saddle-node bifurcation with the transition state TS separating the two modes (see Figure 4.1(c,d)) at a finite folding extent $\rho = \rho_c$ where $\rho \equiv \|\boldsymbol{\rho}\|$. The distance ρ_c is given by a competition between rod compression (or bending in alternative models) at the transition state $\sim K\rho^4$, with K a compression modulus, and the spring energy $\sim \kappa\rho^2$; as shown in the Supplementary Note 1, $\rho_c \sim \sqrt{\kappa/K}$. Other choices of κ_i can eliminate the other mode.

Thus joint stiffnesses change the topological connectivity of undesired modes in state space; see Figure 4.1d. As a result, the undesired mode can be made inaccessible from the flat state, which now continuously connects with only the desired mode. If the network is actuated slowly relative to relaxation timescales of the stiff joints, the network will fold into the desired mode and stay in that state even for $\rho > \rho_c$, despite the reappearance of the undesired mode at finite ρ.

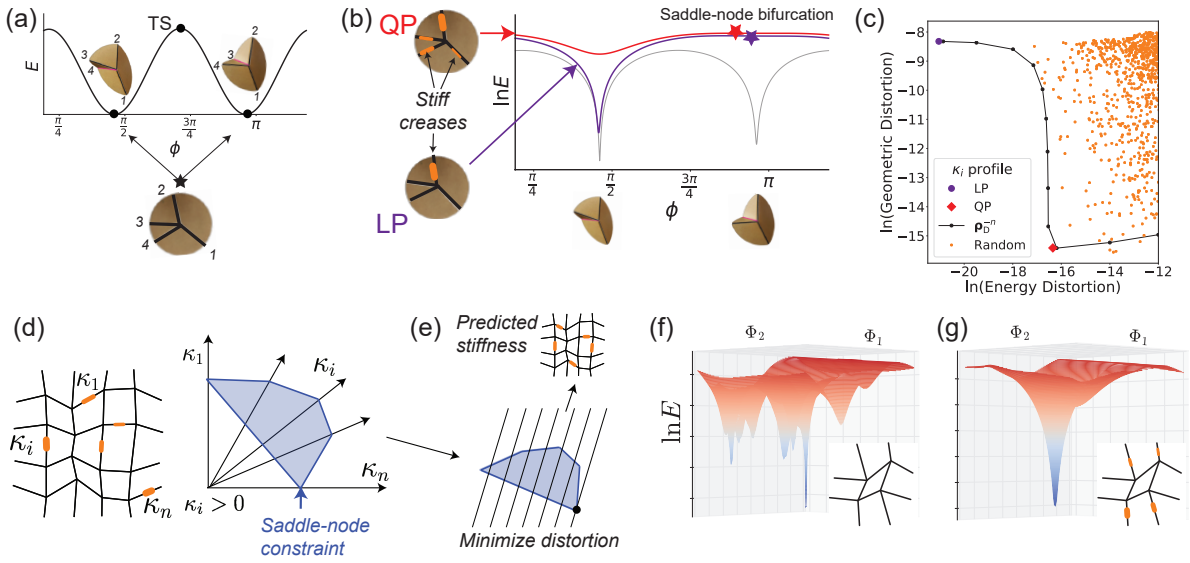

Figure 4.2: Heterogeneous stiff creases can simplify the landscape of self-folding sheets near the flat state

(a) An origami 4-vertex has a choice between two distinct folding modes at the flat state (ϕ - null space angle variable, see Supplementary Note 2). (b) Stiff creases completely eliminates a chosen mode by combining it with a nearby transition state (TS) in a saddle-node bifurcation (Thickness of orange strip indicates stiffness.) (c) Trade-off: Stiff creases distort the desired mode while eliminating undesired modes. Stiffness profiles that minimize energy distortion (e.g., Linear Programming (LP) method) cause large geometric distortion and vice-versa (e.g., Quadratic Programming (QP) method). (d) The exponentially many misfolding modes of large sheets are all eliminated if the stiffness profile κ_i satisfies a linear constraint, shown here as a simplex. (e) We can minimize distortion (energy or geometry) of the desired mode by optimizing crease stiffness on this simplex. (f-g) All but one chosen minimum in a pattern's energy landscape (at small overall folding) can be eliminated by stiff creases predicted by the procedure in (e).

4.1.2 Misfolding in self-folding sheets

Self-folding sheets (or self-folding origami) are structures programmed to have one unique low or zero energy mode [9, 10, 11]. However, self-folding sheets, even when programmed with a single zero energy mode, have been shown to have exponentially many undesirable misfolding modes accessible from the flat state [22, 81]. We show how crease stiffness can change the topological connectivity of these modes and leave only the desired folding mode accessible from the flat state.

To solve the misfolding problem for diverse folding forces, our approach intentionally ignores external folding forces when reprogramming the topological connectivity of modes. Since folding success relies on the bifurcation diagram topology, our results are mathematically robust to several classes of folding forces as shown later.

4.1.3 Avoided bifurcation in a 4-vertex

The atomic unit of self-folding origami is a 4-vertex [41]. Much like the 4-bar linkage, the 4-vertex has one degree of freedom but the flat unfolded 4-vertex is at a branch point, a meeting point of two distinct folding motions [12, 77], distinguished by the Mountain-Valley states of the creases (Figure 4.2(a)) [78, 79]. These two motions are shown as zero energy minima in Figure 4.2(a) using a model of vertex energy presented in the Supplementary Note 2, with a transition state TS separating them. This binary choice is the origin of the exponentially many misfolds of large self-folding sheets.

As with the 4-bar linkage, we wish to lift and eliminate one of the two folding motions, making it inaccessible from the flat state. We introduce stiffness at the creases modeled as a torsional spring with $\rho = 0$ rest angle and energy $E_{\text{Crease},\, i} = \kappa_i \rho_i^2/2$. The energy of the origami vertex is,

$$E = E_{\text{Vertex}} + E_{\text{Crease}} \tag{4.1}$$

where E_{Vertex} accounts for bending of vertex faces [41] and $E_{\text{Crease}} = \sum_i \kappa_i \rho_i^2/2$ accounts for crease stiffness. For details on the energy model see Supplementary Note 2. Crucially, E_{Vertex}

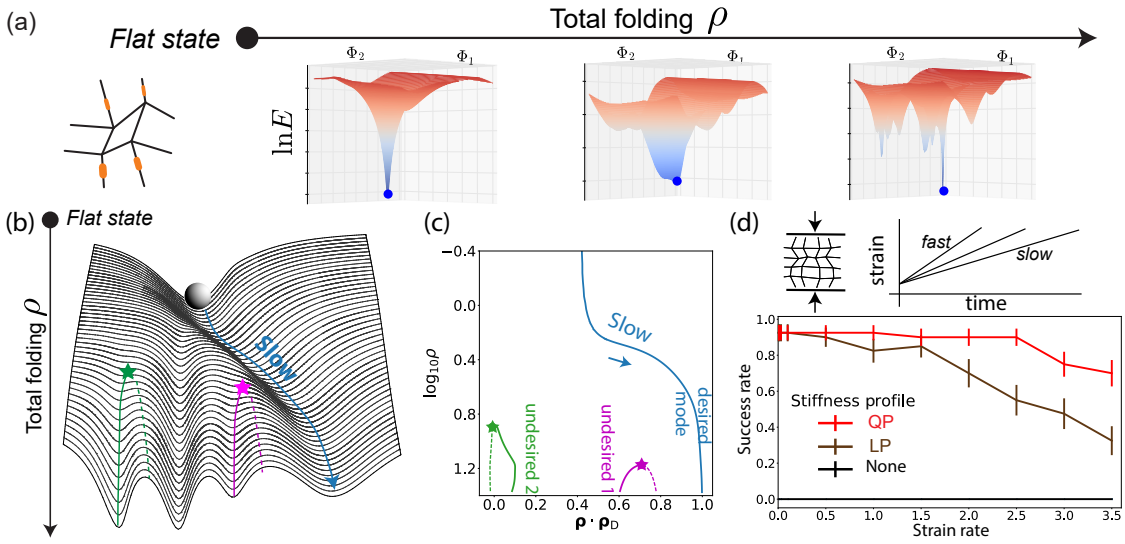

Figure 4.3: Stiff creases change the topological connectivity of undesired modes and promote folding at slow speeds
(a) The energy landscape of a quad with stiff creases has a unique mode at low strain ρ but becomes more complex at higher strain. (b) However, slow folding will recover the desired state provided the unique mode at low strain is continuously connected to the desired state and does not undergo a saddle-node bifurcation (blue line). Slow folding can be successful even if the unique mode at small strain ρ is quite distorted relative to the desired mode. (c) Bifurcation diagram as a function of total folding ρ for a specific 16-vertex pattern shows select undesired modes (solid lines) being eliminated at bifurcations with saddles (dashed lines). Only the desired mode (blue) survives - albeit distorted - to the flat state. (d) Simulations of folding at a finite strain rate (relative to relaxation timescale of hinges) show high success for slow folding and failures at higher folding speeds (Data from 50 random 16−vertex patterns).

scales with a high power ρ^4 for the two special folding motions.

Let us find the conditions on κ_i for lifting and eliminating a chosen branch - the 'undesired branch' - of the 4-vertex. We assume the folding angles of the undesired mode and the desired mode are $\tilde{\boldsymbol{\rho}}_{\mathrm{U}}$ and $\tilde{\boldsymbol{\rho}}_{\mathrm{D}}$ respectively and that of the transition state TS separating them is $\tilde{\boldsymbol{\rho}}_{\mathrm{TS}}$, all assumed to be defined near the flat state and normalized (with unity magnitude). See Figure 4.2(a).

Let $E_{\mathrm{TS}}(\rho)$ be the energy of TS at some chosen total folding $\rho \equiv \|\boldsymbol{\rho}\|$. As the vertex null space (at fixed ρ) is 1-dimensional and compact [22], these features ($\tilde{\boldsymbol{\rho}}_{\mathrm{U}}$, $\tilde{\boldsymbol{\rho}}_{\mathrm{D}}$, $\tilde{\boldsymbol{\rho}}_{\mathrm{TS}}$ and E_{TS}) can all be computed numerically efficiently using peak analysis. Here, we will focus on eliminating the undesired minimum up to a distance ρ_{c} from the flat state and return to larger folding behaviors later. To lift and eliminate the undesired minimum, we should choose a heterogeneous stiffness

profile that raises the undesired mode more than the transition state TS. This constraint - requiring a saddle-node bifurcation - can be written as,

$$\frac{1}{2}\rho_c^2 \sum_{i \in \text{creases}} \kappa_i \left[(\tilde{\rho}_U)_i^2 - (\tilde{\rho}_{TS})_i^2 \right] \geq E_{TS} \tag{4.2}$$

In addition, all crease stiffnesses must be non-negative:

$$\kappa_i \geq 0 \tag{4.3}$$

Note that both constraints are linear in the stiffnesses κ_i.

Any set κ_i satisfying the above constraints that predominantly raises the undesired mode will eliminate it in a saddle-node bifurcation at a total folding distance ρ_c, making it inaccessible from the flat state.

Only the desired mode is stable in the neighborhood of the flat state but it can be significantly distorted by the stiff creases. As shown in Figure 4.2, with stiff creases, the desired mode is of non-zero energy ('Energy distortion') and can also have distorted folding angles ('Geometric distortion', defined by one minus the normalized dot product of the desired mode and the obtained minimum). We wish to formulate design principles for choosing stiffness profiles κ_i, consistent with the above constraints, that best facilitate designed folding motions.

We devise two design strategies: (1) Minimizing energy of the desired mode (Energy optimization), (2) Minimizing geometric distortion of the desired mode (Geometric optimization). We find that different crease stiffness profiles generally trade-off energy and geometric distortion.

Energy optimization is simple: the desired mode has non-zero energy $E(\boldsymbol{\rho}_D) = \sum \kappa_i (\rho_D)_i^2/2$ because of crease stiffness. As this function is linear in κ_i, optimization subject to the saddle-node constraints Equations (4.2-4.3) is equivalent to a Linear Programming (LP) problem [107, 108]:

$$
\begin{aligned}
\underset{\kappa_i}{\text{minimize}} \quad & E(\boldsymbol{\rho}_{\mathrm{D}}) = \frac{1}{2}\sum_i (\rho_{\mathrm{D}})_i^2 \kappa_i \\
\text{subject to} \quad & \rho_{\mathrm{c}}^2 \sum_i \left[(\tilde{\rho}_{\mathrm{U}})_i^2 - (\tilde{\rho}_{\mathrm{TS}})_i^2\right]\kappa_i \geq 2E_{\mathrm{TS}} \,, \\
& \kappa_i \geq 0, \;\; i \in \text{creases} \,.
\end{aligned}
\tag{4.4}
$$

Linear Programming problems are solved in polynomial time, as long as an efficient algorithm is used. Further, the optimal stiffness profile κ_i is generically sparse. In a 4-vertex, only one crease needs to be stiff.

Geometric distortion is minimized if fold angles in the surviving minimum with stiff creases closely corresponds to the fold angles $\boldsymbol{\rho}_{\mathrm{D}}$ of the desired branch without stiff creases. Here, we use the gradient of the energy with stiff creases, but evaluated at ρ_{D}, as a proxy for such geometric distortion. As shown in the Supplementary Note 4, this proxy, after projecting out the component of the gradient in the ρ_{D} direction, is $F_{\mathrm{QP}} = \rho_{\mathrm{D}}^2 \sum_{i\in\text{creases}} \kappa_i^2 (\rho_{\mathrm{D}})_i^2 \; - \sum_{i,j} \kappa_i \kappa_j (\rho_D)_i^2 (\rho_{\mathrm{D}})_j^2$. F_{QP} is a positive semi-definite quadratic function of κ_i. Optimization of F_{QP} - with the linear constraints in Equations (4.2-4.3) - is facilitated by efficient Quadratic Programming (QP) algorithms.

In practice, the LP and QP prescriptions do well at optimizing their respective strategies (i.e. energy and geometry) for a single vertex. Figure 4.2(c) shows how these prescriptions indeed do better than choosing random stiffness profiles that satisfy the constraints. The black line $\kappa_i \sim (\rho_{\mathrm{D}})_i^{-n}$ for positive n, shows that stiffness profiles trade-off energetic and geometric distortion.

4.1.4 Stiffness profiles in large self-folding sheets

Large origami patterns have exponentially many distractor minima states, making them near impossible to fold correctly [22, 81]. Still, the ideas of the previous section can be used to lift all but one of these minima at small folding angles. Crucially, the desired self-folding motion of a large pattern [21, 43] is consistent with exactly one of the two folding modes for each of its constituent 4-vertices. Thus, for a pattern with V vertices, the saddle-node constraint in Equation (4.2)

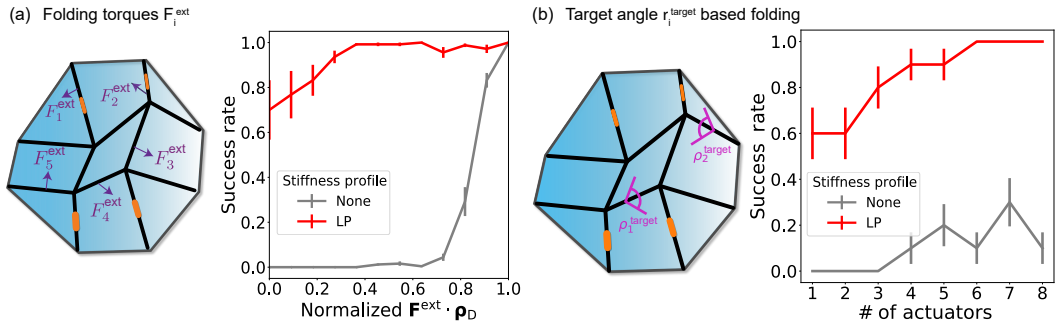

Figure 4.4: Sheets with stiff creases dramatically improve folding for a wide range of external forces applied to specific creases
We consider folding by (a) different folding torques F_i^{ext} or (b) springs that target desired fold angles ρ_i^{target} for different creases i. (a) Forces F_i^{ext} are much more likely to fold into the desired mode with stiff creases (red data) (predicted by linear programming) than with freely folding creases (gray). Even folding forces F_i^{ext} very poorly aligned with the desired pathway $(\rho_D)_i$ result in the desired pathway (Averaged over 5 random 16-vertex patterns, 100 random F_i^{ext} for each data point). (b) Patterns folded using springs of given target angle ρ_i^{target} on select random creases i. Successful folding into the desired mode is dramatically improved in patterns with stiff (LP predicted) creases compared to free folding creases. Even a single actuator is successful a significant fraction of the time (Data averaged over 10 random patterns).

generalizes to V linear constraints, one for each vertex v:

$$\rho_c^2 \sum_{i \in \text{creases of } v} \kappa_i \left[(\tilde{\rho}_{U,v})_i^2 - (\tilde{\rho}_{TS,v})_i^2 \right] \geq 2E_{TS,v}. \tag{4.5}$$

Note that the constraints are dependent since vertices share creases. These linear constraints, along with $\kappa_i > 0$, define a simplex in the space of crease stiffnesses as shown in Figure 4.2d. We can still use LP and QP algorithms as before to find optimized stiffness profiles.

4.1.5 Larger folding angles and adiabatic folding

Figure 4.2(f-g) shows that applying a LP stiffness profile to a quad pattern lifts all but one minimum close to the flat state. However, we noticed that folding beyond a certain angle gives rise to many new minima (Figure 4.3a). To understand this, note that the saddle-node bifurcation constraint,

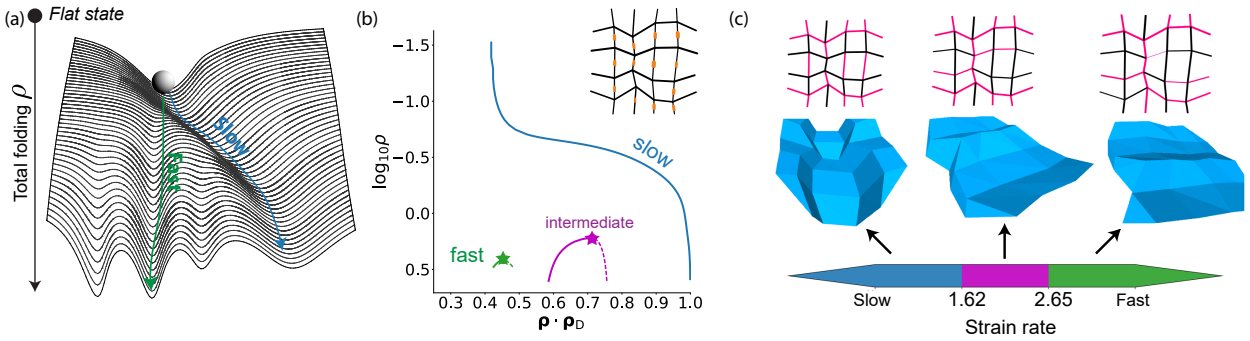

Figure 4.5: Folding speed can controllably select between different folding pathways (a) While slow folding follows the continuous deformation of the unique mode at low strain ρ (blue), fast folding results in a state that most 'resembles' that low-ρ mode (green). If the unique low-ρ mode is significantly distorted in geometry relative to the slow folding target, slow and fast folding can result in very different outcomes. (b-c) We systematically attempted folding at different strain rates (relative to a fixed hinge relaxation timescale) for the 16-vertex pattern with stiff creases shown. We find three distinct outcomes at slow, intermediate, fast rates that completely differ in their Mountain-valley states, geometry and energy. The slow folding outcome corresponds to following the blue path in (b) while the intermediate and fast pathways cross over from blue to the magenta and green modes respectively at some intermediate folding angles.

Equation (4.2), only ensures the absence of undesired modes up to a total folding ρ_c at which $E_{TS}(\rho_c)$ is computed. Intuitively, crease stiffness ($\sim \rho^2$) becomes less important than face bending ($\sim \rho^4$) as folding proceeds and undesired modes are restored in a series of saddle-node bifurcations.

At first sight, the reappearance of undesired modes at large ρ might seem disappointing. However, if folding is carried out adiabatically - i.e., slowly relative to hinge relaxation timescales - these modes do not impact folding at all. Adiabatic folding, by definition, will follow the continuous deformation of the unique low-ρ minimum (blue paths in Figure 4.3(b-c)), even if it is significantly distorted relative to the desired state. Undesired states, on the other hand, are not continuously connected to the low-ρ mode.

To test whether our stiff crease prescriptions are able to consistently create such adiabatic pathways, we sampled 50 random patterns, each with a programmed low-energy motion using the loop equations of [21, 43]. Such patterns have $\sim 10^3$ higher energy undesired modes [22] and thus folding almost always fails (Figure 4.3d).

We then augmented the sampled patterns with stiff creases resulting from LP and QP prescrip-

tions and simulated folding at varying speeds. In simulations, we assume the crease hinges follow a first order equation with a relaxation timescale τ_{relax}; this timescale is known to vary with material implementation [109]. These stiff patterns achieve a success rate in excess of 90% when folded slowly (Figure 4.3(d)), compared with the expected $< 0.1\%$ success rate with free folding creases. Thus our stiffness heuristics are useful for slow folding, yet imperfect.

The small fraction of failed cases represent patterns where the unique low-ρ mode and the desired high-ρ mode undergo distinct saddle-node bifurcations at intermediate ρ and thus do not connect up. Such bifurcations are mathematically forbidden if these states are the lowest energy states for all ρ. Complex optimization methods that account for details of non-linear energy landscape at all intermediate ρ might be able to better protect from such bifurcations. However, we find that simple heuristics, e.g., based on the energy of low and high-ρ states alone, are sufficient to protect the adiabatic pathway from bifurcations for complex patterns. See Supplementary Note 5 for more analysis of failures.

4.1.6 External folding forces applied to creases

Our crease stiffness prescriptions are meant to eliminate undesired modes in the intrinsic energy landscape of a sheet and not just for a particular model of folding - hence no particular folding forces are assumed. Once undesired modes are eliminated, many typical classes of folding forces cannot re-introduce such modes near the flat state.

Besides strain-controlled folding tested above, another important class of folding forces [10, 13] involves folding torques F_i applied to specific creases i; see Figure 4.4. A related method involves setting target folding angles ρ_i^{target} for particular creases (see Supplementary Note 3). Near the flat state, both of these methods change the energy landscape by a linear tilt ($\sim F_i\rho_i$). Mathematically, such tilts cannot create new undesired minima close enough to the flat state. We tested folding success in these methods of actuation as a function of the number of actuated creases. Folding success is enhanced by orders of magnitude due to the stiff creases predicted here as shown in Figure 4.4. Hence our approach to modifying the topology of the bifurcation diagram is also

useful when external folding forces are present – in fact, such a modification is necessary for successful folding.

Earlier works [77] have tried to find such specific folding torques or folding springs to fold along a desired mode. Mathematically, such approaches appear similar to ours since they both involve quadratic modifications to the energy function. However, our crease stiffness is a quadratic potential centered at the branch point (i.e., the flat state) and hence is able to modify the topology of that point successfully while springs with finite target angles [77] are effectively linear tilts at the branch point. As noted in [77], successfully folding the pattern using such a method requires undesired branches to have negative dot products with the desired branch, a very unlikely scenario for larger patterns. In contrast, our quadratic potentials at the flat state face no such restriction and thus work in a dramatically larger context.

Our approach is also different in practice. Prior approaches [22, 77] sought sheets with freely folding creases that must be carefully actuated using calculated folding forces. Our approach designs sheets with calculated crease stiffness profiles that can be carelessly actuated.

4.1.7 Folding speed-dependent target structures

We have seen that the unique low-ρ minimum funnels adiabatic folding to the desired state in a glassy landscape, even if the unique low-ρ mode is significantly distorted relative to it. However, the success rate drops with folding rate; see Figure 4.3d.

Such a drop in success rate is to be expected since very fast folding essentially takes the pattern from the unique low-ρ state to high-ρ with quenched geometry and then relaxes to the nearest minimum. Thus, as suggested by Figure 4.5a, fast folding from the unique low-ρ minimum reproducibly picks the folded configuration with closest geometric resemblance to the low-ρ minimum.

These considerations suggest an intriguing possibility - programming the bifurcation diagram using stiff creases can program different folding pathways that are followed at different speeds of folding.

We tested this hypothesis on a 16-vertex pattern with LP springs whose unique low-ρ mode has

significant geometric distortion relative to the adiabatic folding outcome; see Figure 4.5b. We systematically folded this structure at increasing speeds relative to its hinge relaxation timescale. We find three completely distinct but reproducible folded structures in regimes of slow, intermediate and fast folding; see Figure 4.5c.

4.2 Discussion

In this paper, we argued that meta-materials design should be conceptualized as targeted design of an entire dynamic pathway that avoids undesired behaviors, and not just target a desired final state. We showed how such pathways and their topological connectivity can be programmed by controlling the bifurcation diagram; we applied our method to remove the exponentially many misfolding motions intrinsic to self-folding origami.

We showed that the bifurcation diagram can be modified by stiff joints, an inevitable feature of most experimental realizations of origami, linkage networks and other meta-materials. Thus, our proposal is conservative - it does not require specific directional information at hinges [10], temporal staging [48] or using non-flat sheets [41]. Our general approach applies to any other heterogeneous bulk imperfection that couples to different folding modes unequally.

A particularly intriguing direction suggested by our work is the ability to geometrically program different behaviors at different speeds. These outcomes can have independently tuneable mechanical properties, such as energy absorption [21]. While such complex speed-dependent phenomena are actively studied in materials (e.g., cornstarch [110, 111]), our approach suggests that speed-dependent behaviors can be dictated simply by the bifurcation geometry of the meta-material.

4.3 Supplementary Figures

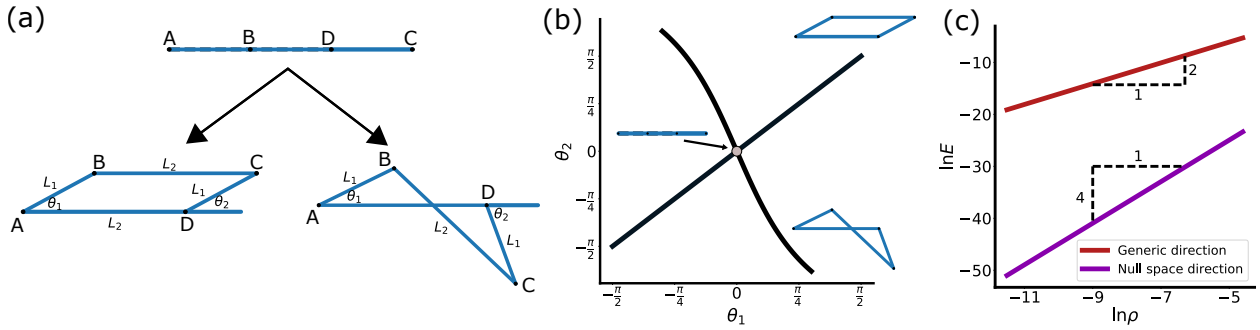

Figure 4.6: Modes of the 4-bar linkage
(a) A stiff parallelogram type 4-bar linkage bifurcates from the top 'flat' state into two distinguishable motions, (b) The two motions are defined by the relations imposed on angles θ_1, θ_2, (c) The two special zero-energy motions span a linearized null space, configurations in which scale quartically with the distance form the flat state $E \sim \rho^4$. In contrast, random configurations of the linkage have an energy that scales quadratically with this distance $E \sim \rho^2$.

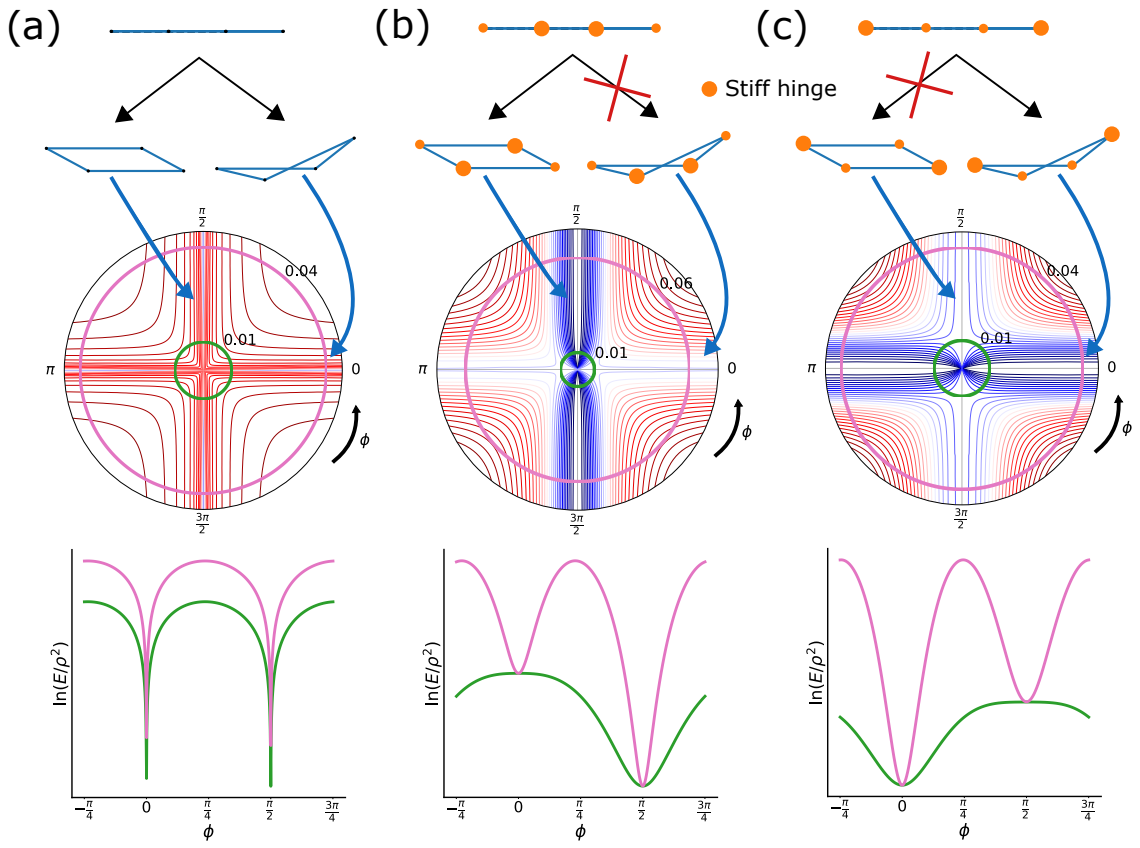

Figure 4.7: Each of the linkage modes can be lifted with proper hinge stiffness values (a) With freely rotating hinges, the two zero-modes bifurcate at the flat state in the center of the landscape diagram, whose radius corresponds to ρ, and angle ϕ is the mixing angle of the two zero-modes spanning the linearized null space (contours represent scaled energy E/ρ^2), (b) Choosing the two hinges denoted by large orange circles to be stiffer than the other two, the right mode is lifted, making only the left mode continuously accessible from the flat state, shown as a continuous blue valley starting at $\rho = 0$. The right mode is regained at a larger radius $\rho > \rho_c$, as seen in the contour curvature, (c) By making the other two hinges stiffer, the left mode is lifted near the flat state.

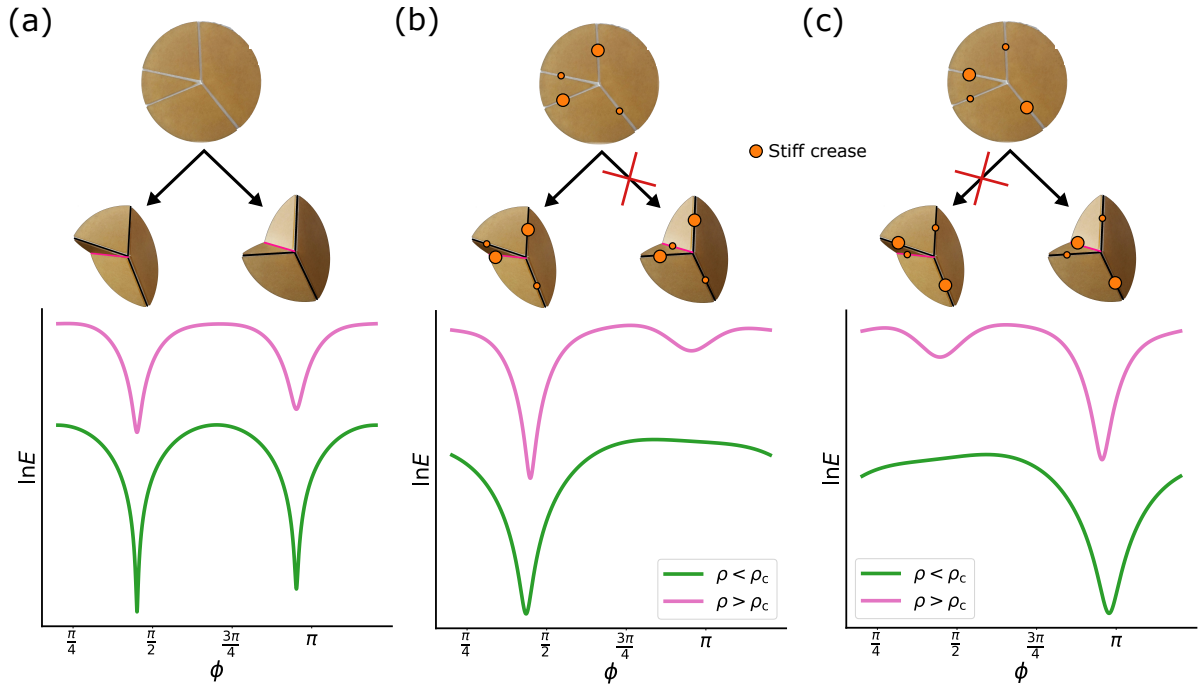

Figure 4.8: Lifting modes in origami 4-vertices
Just like in the 4-bar linkage example, each of the two zero-modes of the origami 4-vertex can be lifted given a proper crease stiffness profile. (a) The modes of a vertex with perfectly soft creases bifurcate at the flat state, so that two minima appear in the linearized null space for any value of the folding magnitude ρ (ϕ is the mixing angle of two vectors spanning the linearized null space), (b) The heterogeneous crease stiffness described here, with stiffer creases denoted by larger orange circles, lifts the right mode, so that only the left mode is continuously accessible from the flat state (the right mode is again regained at larger folding values $\rho > \rho_c$), (c) Changing the crease stiffness profile as shown, we can instead lift the left mode near the flat state.

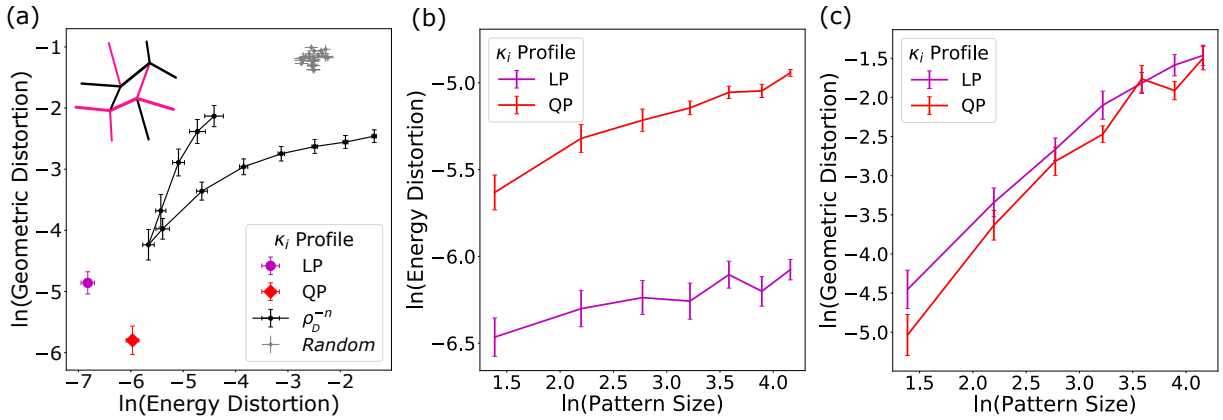

Figure 4.9: Performance of Linear and Quadratic Programming stiffness selection protocols for large origami patterns
a) LP and QP stiffness optimize energetic and geometric distortion of the designed folding motion for quad patterns. LP and QP stiffness profiles improve these distortions by orders of magnitude compared to random profiles. b) Larger patterns with many vertices are harder to optimize, with designed motion energy scaling as soft power laws with system size. c) Geometric distortion grows with system size.

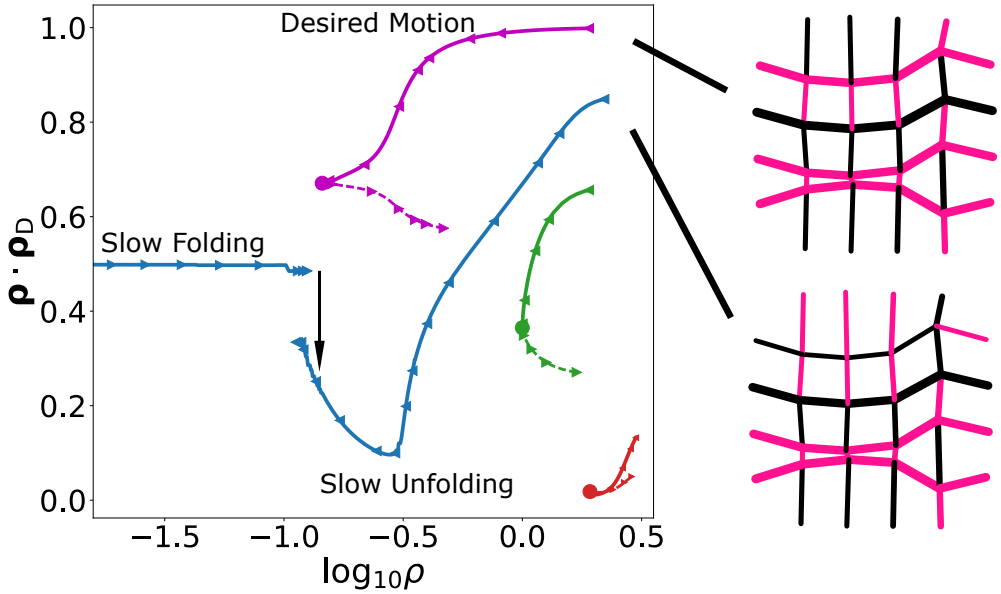

Figure 4.10: A pattern for which slow folding with LP stiffness profile fails
The low-ρ unique minimum, rather then continuously connecting to the desired state, terminates in a saddle-node bifurcation and discontinuously jumps to a distractor folded state.

4.4 Supplementary Notes

Supplementary Note 1 - 4-bar linkage

The 4-bar linkage [102, 103] discussed in the main text is composed of 2 pairs of rods with lengths L_1, L_2 (Supplementary Figure 4.6(a)). For simplicity, let us assume that points A and D are held fixed in space with distance L_2 between them. Points B and C are allowed to move in a plane, with an energy cost associated with changing the lengths of the three rods from their relaxed state $E = \frac{1}{2}K\Delta L^2$ (alternatively, energy cost can arise from bending the three rods from their relaxed straight configuration).

This system is a one degree-of-freedom mechanism, as there are 3 constraints (rod lengths) imposed on 4 variables (x, y positions of nodes B, C). As such, configurations satisfying the constraints define one-dimensional motions. These particular cases are fully described in terms of the angles θ_1, θ_2 between rods $L_{\mathrm{AB}}, L_{\mathrm{CD}}$ and the x-axis. When accounting for the non-linearity of the constraints, one finds two one-dimensional solutions (Supplementary Figure 4.6(b)) for the system motions given by [95]

$$\begin{aligned} \theta_2^{\alpha} &= \theta_1 \\ \theta_2^{\beta} &= \cos^{-1}\Big\{\frac{[1+(\frac{L_2}{L_1})^2]\cos\theta_1 - 2\frac{L_2}{L_1}}{[1+(\frac{L_2}{L_1})^2] - 2\frac{L_2}{L_1}\cos\theta_1}\Big\} \end{aligned} \tag{4.6}$$

The two solutions share a configuration $\theta_1 = 0 \rightarrow \theta_2 = 0$. Consider this special 'flat state' $\theta_1 = \theta_2 = 0$, for which the node positions are $\mathbf{P}_0 = [x_{\mathrm{B}} = L_1, y_{\mathrm{B}} = 0, x_{\mathrm{C}} = L_1 + L_2, y_{\mathrm{C}} = 0]$. Let us denote the distance of a configuration from the flat state by $\rho \equiv \|\boldsymbol{P} - \boldsymbol{P}_0\|$. When constraints are allowed to be broken by compression (or bending) of the rods, allowing nonzero energy configurations in the full 4-dimensional $\boldsymbol{\rho}$-space, the configuration energy generically scales as $E \sim K(\boldsymbol{P} - \boldsymbol{P}_0)^2 \sim \rho^2$. However, close to the flat state the two special zero-energy motions

span a linearized null space in which the energy scales more softly $E_{\mathrm{NS}} \sim K(\boldsymbol{P} - \boldsymbol{P}_0)^4 \sim \rho^4$ (Supplementary Figure 4.6(c)). To see this, note that to lowest order in ρ, the two zero-energy motions are composed of just vertical motions of points B, C:

$$\boldsymbol{v}_{\pm} = \frac{1}{\sqrt{2}}[0, 1, 0, \pm 1]$$

These vertical motions are small compared to rod lengths, and the Pythagorean theorem tells us that the energy difference due to rod extension vanishes to quadratic order. Figure 4.1 shows the energy of configurations in the null space for given magnitudes of the vector $\boldsymbol{P} - \boldsymbol{P}_0$.

As the null space energy of the system scales like ρ^4, we show that placing heterogeneous quadratic torsional springs on the hinges raises the energy of one motion more than the other. This facilitates the removal of the flat state bifurcation, and the mechanical preference of just one of the two motions. Let us first define the energy model for the 4-bar linkage:

$$E(\boldsymbol{\rho}) \equiv \frac{1}{2}K \sum_{i \in \text{bars}} \Delta L_i^2 + \frac{1}{2} \sum_{j \in \text{hinges}} \kappa_j \theta_j^2, \tag{4.7}$$

With torsional springs κ_j attached to each of the four hinges, such that the springs are relaxed at the flat state. With stiff hinges, the energy of all configurations now scale as ρ^2 (including those in the linearized null space). However, as different hinges have different stiffness values, it is possible to raise the energy of different configurations to differing extents. If all hinges are free (Supplementary Figure 4.7(a)), we again have the two zero-modes established before. The middle row shows the scaled null space energy E/ρ^2, with radius ρ and angle ϕ is the mixing angle of the two zero-modes spanning the 2-dimensional null space:

$$\boldsymbol{\rho} = \rho(\boldsymbol{v}_+ \sin\phi + \boldsymbol{v}_- \cos\phi)$$

The energy is scaled by ρ^2 so that the low energy paths in the landscape are clearly visible in the contour topography as blue valleys between red ridges.

Let us now choose a heterogeneous hinge stiffness profile as seen in Supplementary Figure 4.7(b). Here, large circles correspond to stiff hinges with stiffness values $\kappa_1 = 10^{-4}K$, while small circles denote softer hinges with $\kappa_2 = 10^{-6}K$. With this choice we find that one of the original zero-modes is lifted up to a distance from the flat state $\rho_c \sim \sqrt{\kappa_1/K} = 10^{-2}$. At $\rho \sim \rho_c$, the lifted mode returns by a saddle-node bifurcation, and is clearly seen as a stable minimum in larger radii. By switching the locations of the stiff and soft hinges, we find that the other zero-mode is lifted near the flat state (Supplementary Figure 4.7(c)). As in the other choice of hinge stiffness value, the lifted zero mode returns via a saddle-node bifurcation at $\rho \sim \rho_c$.

In general, when heterogeneous stiffness profiles are applied to the hinges, a critical crossover value arises $\rho_c \sim \sqrt{\tilde{\kappa}/K}$ ($\tilde{\kappa}$ the dominant hinge stiffness scale), where the stiff hinge energy is comparable to the intrinsic energy. We find that one minimum exists in the energy landscape at distances $\rho \ll \rho_c$. The unique minimum corresponds to one of the two zero-modes of the original system, and continuously connects to it when ρ is increased. The other zero-mode of the system is always regained at $\rho \sim \rho_c$ via a saddle-node bifurcation. If a random heterogeneous stiffness profile is chosen, the zero-mode that survives close to the flat state is decided by which pair of hinges is stiffer, as described by Supplementary Figure 4.7(b-c).

Finally, given a fluctuation energy scale ϵ, effectively removing the bifurcation requires the energy barriers when the second minimum is regained to be larger than ϵ. This requirement establishes a lower bound on the hinge stiffness $\tilde{\kappa} > \sqrt{\epsilon K}$.

Supplementary Note 2 - Origami energy model

4-Vertex energy

Any origami vertex embedded in 3-dimensional space gives rise to 3 constraints relating the dihedral angles by which the different creases fold (i.e. angles between faces) [37, 42]. These constraints are derived for any vertex by requiring that the vertex faces do not bend when folded. Consider a small disk surrounding the vertex, whose configurations is defined by the folding angles

ρ_i and the in-plane angles between the creases α_i. If the folded vertex faces do not bend anywhere, it is possible to trace the edge of the disk around the vertex and return to the starting point using simple 3d rotations. The tracing motion around the vertex consists of alternating rotations with the circular section angle about the central axis of the vertex α_i, and then rotating about the dihedral angle ρ_i. This is done for each crease, until one returns to the original position on the vertex. If one orients the current vertex face such that the face occupies the xy-plane and the crease is on the x-axis, these two rotation matrices are given by

$$R_i = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\rho_i & -\sin\rho_i \\ 0 & \sin\rho_i & \cos\rho_i \end{pmatrix} \begin{pmatrix} \cos\alpha_i & -\sin\alpha_i & 0 \\ \sin\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{4.8}$$

In general $R_i = A_i B_i$ where A_i is a rotation matrix about an axis along crease i by angle ρ_i, while B_i is a rotation matrix about an axis perpendicular to face i by angle α_i.

The condition that the vertex faces do not bend becomes

$$\prod_i R_i = I, \tag{4.9}$$

where the product is taken over all creases and faces i, and I is the 3×3 identity matrix. Supplementary Equation (4.9) can be shown to be equivalent to 3 independent equations for the off-diagonal upper triangle of the matrix. Crucially, these are 3 non-linear constraints relating the folding values ρ_i around the vertex. The 4-vertex is a one degree-of-freedom object, as 3 equation relate its 4 folding angles.

Similarly to the linkage discussed above, the fact that an origami 4-vertex can only support one-dimensional motions does not imply that only one such motion exists. A single 4-vertex is known to have two zero-energy folding motions that meet at the flat, unfolded state. These special motions exactly satisfy the three vertex constraints.

Near the flat state, all $\rho_i \approx 0$, and the matrices of equations (4.8-4.9) are approximately $2d$ rotations about an axis perpendicular to the flat vertex. One vertex constraint thus becomes degen-

erate, and a linearized 2d null space emerges. The preceding considerations apply to all vertices constructing the pattern.

The constraint equations can be expanded about the flat state $\boldsymbol{\rho} = 0$:

$$T_a(\boldsymbol{\rho}) = C_a^i \rho_i + D_a^{ij} \rho_i \rho_j + \dots \tag{4.10}$$

We take the energy of the vertex to be,

$$E_{\text{Vertex}} = \frac{1}{2} \kappa_{\text{f}} \sum_a T_a^2$$

with κ_{f} a face bending modulus [43, 39, 14, 81]. Thus our energy is a measure of the violation of vertex constraints in Supplementary Equation 4.9. For thin sheets, violation of vertex constraints manifests as bending of vertex faces since stretching is energetically expensive compared to bending [54].

With this definition of vertex energy, we find that configurations $\vec{\rho}$ that violate the linear constraints Supplementary Equation 4.9, i.e., $C_a^i \rho_i \neq 0$, are characterized by an energy $E \sim \rho^2$. On the other hand, configurations $\boldsymbol{\rho}$ in the $2d$ linearized null space satisfy $C_a^i \rho_i = 0$, thus their energy scales as $E \sim \rho^4$ [22] (The two special zero-energy motions of the vertex satisfy the constraints to all orders in ρ).

These two distinct scaling laws allow us to avoid the flat state bifurcation by lifting one of the zero-modes, in a similar manner to the way it was done for the 4-bar linkage. If the creases themselves have a heterogeneous profile of stiffness (or bending rigidity), some folding configurations can be made energetically favorable compared to others. This last notion is encoded in the energy model of the vertex

$$E = E_{\text{Vertex}} + E_{\text{Crease}} = \kappa_{\text{f}} T_a^2 + \frac{1}{2} \sum_{i \in \text{creases}} \kappa_i \rho_i^2, \tag{4.11}$$

with κ_i the stiffness of crease i. In practical applications, the folding stiffness moduli of the creases κ_i will depend on each crease's thickness, length and material properties. Generally the length and material are given by the type of application, so that the value of κ_i can be set by proper choice of the crease thickness t. It is known that for general elastic materials $\kappa \sim t^3$ [55].

Supplementary Figure 4.8 shows how changing the crease stiffness profile κ_i allows us to choose which of the two modes is lifted near the flat state, while the scale of the crease stiffness sets the folding value $\rho_c \sim \sqrt{\tilde{\kappa}_i/\kappa_f}$ in which the lifted mode returns via a saddle-node bifurcation. As we had for the 4-bar linkage, the linearized null space is spanned by two vectors $\boldsymbol{v}_1, \boldsymbol{v}_2$, such that configurations in the null space are given by

$$\boldsymbol{\rho} = \rho(\boldsymbol{v}_1 \sin\phi + \boldsymbol{v}_2 \cos\phi).$$

Large patterns with loops

When 4-vertices form a closed loop, an extra constraint is defined by the requirement that folding angles are consistent around each loop [21]. Just as for the independent vertex constraints, the extra constraint can be broken at the expense of bending the stiff face encompassed by the loop. We model this bending by adding a stiff 'face' crease inside each loop of vertices [22]. Since both vertex and loop constraints are broken by bending the same stiff faces, the bending modulus is the same κ_f. Together with the heterogeneous stiff creases defined in the main text, the origami energy model is given by [43, 39, 14, 81]

$$\begin{aligned} E_{\text{sheet}}(\boldsymbol{\rho}) &\equiv E_{\text{Vertex}} + E_{\text{Loop}} + E_{\text{Crease}} \\ &= \kappa_f \sum_a T_a^2 + \frac{1}{2}\kappa_f \sum_{f\in\text{faces}} \rho_f^2 + \frac{1}{2} \sum_{i\in\text{creases}} \kappa_i \rho_i^2, \end{aligned} \tag{4.12}$$

We note that in the linearized null space about the flat state, both the vertex and loop terms scale as ρ^4 , while the crease bending term generically scales as ρ^2. Thus, at small folding magni-

tudes $\rho \ll \rho_c \sim \sqrt{\bar{\kappa}_i/\kappa_f}$ bending of creases is expensive and suppressed - minimizing the convex quadratic energy term. In contrast, at large folding magnitudes $\rho \gg \rho_c$, face bending becomes expensive, such that all bending occurs at the creases. We conclude that by choosing a crease stiffness profile appropriately, the flat state branching can be avoided for arbitrarily large self-folding origami patterns.

Supplementary Note 3 - Folding methods

Patterns with stiff creases as defined above will settle to the flat state, the global minimal energy configuration of E_{sheet} (Supplementary Equation 4.12). When external folding torques $F_i^{\text{ext}}(\rho, t)$ are applied to the creases i across the pattern, the sheet is folded away from the flat state by balancing the external forces and the internal forces due to face and crease bending (Supplementary Equation 4.12). We define folding through the equation,

$$\tau_{\text{relax}} \frac{d\rho_i}{dt} = -\frac{\partial E_{\text{sheet}}(\boldsymbol{\rho})}{\partial \rho_i} + F_i^{\text{ext}}(\rho, t) \tag{4.13}$$

Here, τ_{relax} is the folding relaxation timescale of the creases, determined by the elastic properties of the material [10]. This critical parameter sets the relevant timescale for all dynamics. For example, adiabatic folding implies the folding forces $F^{\text{ext}}(t)$ change slowly relative to τ_{relax} while fast folding implies $F^{\text{ext}}(t)$ change fast relative to τ_{relax}.

In some cases, the folding forces can be captured by an energy function E_{folding}, i.e., $F_i^{\text{ext}}(\rho, t) = -\frac{\partial E_{\text{folding}}(\boldsymbol{\rho})}{\partial \rho_i}$. In these select cases, we can write,

$$\tau_{\text{relax}} \frac{d\rho_i}{dt} = -\frac{\partial (E_{\text{sheet}}(\boldsymbol{\rho}) + E_{\text{folding}}(\boldsymbol{\rho}))}{\partial \rho_i} \equiv -\frac{\partial E_{\text{tot}}(\boldsymbol{\rho})}{\partial \rho_i}.$$

Torque-based folding

Several experiments [10] apply constant folding torques F_i to specific creases. One simple model of such folding experiments would be the following time-dependent energy,

$$E_{\text{tot}} = E_{\text{sheet}} - a(t) \sum_i F_i \rho_i \tag{4.14}$$

Here $a(t)$ describes a time-dependent protocol for ramping up the folding forces F_i. We take $a(t) = vt$, where v measures the speed of folding.

Simulations: We simulated folding using Supplementary Equation 4.14 but without stiff creases and find no success unless F_i has high dot product with the desired mode (Figure 4.4(a)). It is hard to call high dot-product based folding 'self-folding' since such actuation requires a large number of actuator creases, and the torques F_i on each crease need to be tuned carefully.

However, with stiff creases, the 16 vertex pattern folds with large success rates for substantially smaller dot products. For example, a dot product of 0.2 suffices to fold the pattern correctly for $\sim$ 80% of the patterns and forces sampled. The data shown is averaged over 5 patterns and 20 different random torque vectors F_i (for each sample and dot product), folded using a sufficiently slow protocol $a(t)$.

Topologically protected bifurcations:

Figure 4.4(a) shows that sheets with stiff creases have dramatically higher success rate in folding using specific F_i, even though the crease stiffness profiles were chosen without any knowledge of these folding forces. We can understand the reason for success mathematically - our stiff crease prescription attempts to program the bifurcation diagram to eliminate misfolding pathways. Such topological properties of bifurcation diagrams are robust to the kinds of deformations induced by these folding forces.

The folding forces in Supplementary Equation 4.14 above modify the energy by a linear term

$\sum_i F_i \rho_i$. Such modification cannot create new minima near the flat state since E_{sheet} has a saddle-node bifurcation only at a finite ρ_c.

To understand this first in a simple example, consider the function $f(x) + \lambda x$ where $f(x)$ has one unique minimum at some point x_0. In addition to the minimum at x_0, if $f(x)$ was posed at a saddle-node bifurcation at some point x_1 (e.g., $f(x) \sim (x - x_1)^3$ near x_1), then $f(x) + \lambda x$ will develop a new minimum at x_1 for infinitesimal λ. Instead, if $f(x)$ were some finite distance away from a saddle-node bifurcation in a neighborhood of x_1, we are guaranteed that no new minima are created by small enough λ.

In analogy, our design principles eliminated all undesired minima in E_{sheet} at saddle-node bifurcations at some finite distance ρ_c away from the flat state. Thus adding the linear term $\sum_i F_i \rho_i$ cannot create new minima in this landscape close enough to the flat state. Thus we can mathematically anticipate the significant success seen in Figure 4.4(a) with slow folding.

Target angle-based folding

A related method of folding involves target angles ρ_i^{target} for different creases. Such a target-based folding can be modeled by springs with rest angles at ρ_i^{target} that are 'turned on' by some actuation method (e.g., temperature, light changes [13, 10]),

$$E_{\text{tot}} = E_{\text{sheet}} + \frac{1}{2} \sum_i \eta_i (\rho_i - a(t)\rho_i^{\text{target}})^2 \tag{4.15}$$

where $a(t)$ describes the time-dependent protocol by which such creases are actuated; $a(t) = 0$ before actuation and the target angles are ramped up to final target ρ_i^{target}. We take $a(t) = vt$ and simulate slow folding. η_i is the strength of the springs with target angles; only a select set of creases are actuated in this manner ($\eta_i \neq 0$).

Simulations: we applied the folding method of Supplementary Equation 4.15 to 16 vertex patterns. This time, instead of varying the dot product between external forcing and the desired mode,

we vary the number of creases that are actuated using target angle folding (the creases chosen are those who fold most in the desired mode $\boldsymbol{\rho}_D$). When the creases are soft, we again see that this model of folding can rarely find the desired mode (Figure 4.4(b)), even if a few creases are actuated simultaneously. On the other hand, applying the LP stiffness profile to the creases improves the success rate dramatically, even while actuating just 1 or 2 creases (data is averaged over 10 patterns).

Topologically protected bifurcations:

Sheets with designed stiff creases show high success rates with this mode of folding as well. Again, we could have anticipated this success mathematically. Note that Supplementary Equation 4.15 can be expanded and written as,

$$E_{\text{tot}} \sim E_{\text{sheet}} + \frac{1}{2}\sum_i \eta_i \rho_i^2 - a(t)\sum_i \eta_i \rho_i^{\text{target}} \rho_i$$

where we have dropped a term independent of ρ_i. The linear term $a(t)\sum \rho_i^{\text{target}}\rho_i$ is mathematically identical to the linear term in torque-based folding with $F_i = \eta_i \rho_i^{\text{target}}$. Hence folding can be expected to succeed for the same reasons.

However, the quadratic term $\eta_i \rho_i^2$ resembles an additional stiffness for the creases about the flat state, in addition to the designed stiffness κ_i present in E_{sheet}. Such extra stiffness could potentially be problematic. To see this, note that in the main paper, the designed stiffness κ_i were carefully chosen to satisfy a linear lifting condition (or saddle-node bifurcation condition) of the form,

$$\sum_{i \in \text{creases}} R_i \kappa_i \geq E_{\text{TS}}(\rho_c) \tag{4.16}$$

where $R_i \equiv \frac{1}{2}\rho_c^2\left[(\tilde{\rho}_U)_i^2 - (\tilde{\rho}_{TS})_i^2\right]$ and $\tilde{\boldsymbol{\rho}}_U, \tilde{\boldsymbol{\rho}}_{TS}, \rho_c, E_{TS}$ were defined in the main text.

Shifting a solution κ_i of the above equation by $\kappa_i + \eta_i$ could, in principle, violate the above equation. The simulation results in Figure 4.4(b) show that such violations do not occur often for

random patterns. However, when such violation do occur, they would result in undesired minima reappearing.

If η_i are known at the time of design, we can simply account for them in the above lifting constraint and use the following constraint instead,

$$\sum_{i \in \text{creases}} R_i \kappa_i \geq E_{\text{TS}}(\rho_c) - \sum_{i \in \text{creases}} R_i \eta_i \tag{4.17}$$

where $R_i \equiv \frac{1}{2}\rho_c^2\left[(\tilde{\rho}_U)_i^2 - (\tilde{\rho}_{TS})_i^2\right]$.

To summarize, our crease stiffness prescription is designed to solve the misfolding problem for general. Thus we do not account for specific forces in the procedure to determine optimal crease stiffness profiles. In addition to the folding results shown in the main paper, the simulation results in Figure 4.4 show that the predicted stiffness profiles work for a range of folding forces and models, even if the forces were not known at design time. On the other hand, Supplementary Equation 4.17 shows how specific folding forces and methods can be accounted for at design time to change failed cases into successfully folding protocols.

Strain-based folding

In some methods of folding [10], the sheet is generally compressed without specific forces F_i applied to specific creases. Without crease stiffness, such folding without specific forces is almost certain to misfold. We tested the folding success of such folding methods in patterns with stiff creases in simulation.

Patterns are initially set at a small folding magnitude $\rho \ll \rho_c$, and allowed to relax (with fixed ρ) influenced by the sheet potential. As argued above, we find that when heterogeneous crease stiffness profiles are present, there exists a unique minimum at fixed small ρ (up to Z_2 symmetry). The pattern is folded with a strain based algorithm, such that the external torques applied to the

pattern are proportional to the folding angles of the configuration. That is, we use an energy,

$$E = E_{\text{sheet}} - a(t) \sum_i F_i \rho_i, \quad F_i = \frac{1}{2} \rho_i(t) \tag{4.18}$$

In practice, folding the pattern at a specified rate is achieved by an iterative two step algorithm: 1) Increase ρ by a specified amount $\Delta\rho$, keeping the direction $\vec{\rho}$ fixed, 2) Relax the configuration using the origami energy model (keeping ρ fixed).

While the first step in each iteration increases the magnitude of folding, the second step modifies the configuration in the angular directions by gradient descent, finding the nearest angular minimum. The slow folding limit is obtained by setting $\Delta\rho \ll \rho_c$ in step 1, so that the one-dimensional tracks are followed adiabatically. The folding algorithm is implemented using MATLAB constrained optimization functionality. Results on the success of strain based folding methods with heterogeneous crease stiffness profiles are given in Figure 4.3.

Supplementary Note 4 - Crease stiffness selection

In the main text we present general arguments for the choice of crease stiffness profiles. Designed motions in large self-folding origami patterns can be approximately decomposed to individual 4-vertices making a binary choice between their two zero-modes. This idea suggests that the designed motion can be targeted for adiabatic folding by lifting the undesired motion in each 4-vertex independently. The argument gives rise to a set of linear constraints lifting the unwanted minimum at a specific chosen ρ_c value. Together with a positivity constraint, these two inequality constraints limit the choice of stiffness profile, but maintain much of the stiffness design freedom. We note that the simple structure of the constraints guarantees the existence of a large feasible solution space, as the constraints ask for semi-positive $\boldsymbol{\kappa}$ vectors that exist above the hyper-plane defined by Equation 2 of the main text.

Although any random stiffness profile satisfying the inequality constraints will keep only the minimum related to the desired minimum, we further discuss how the design freedom can be

utilized to minimize the energetic or geometric distortion of the resulting motion. These considerations give rise to the Linear and Quadratic Programming methods for selecting the stiffness profile.

Optimizing the energetic distortion of the desired mode ρ_{D} is performed by minimization of its energy due to the stiff creases:

$$E_{\mathrm{Crease}}(\boldsymbol{\rho}_{\mathrm{D}}) = \frac{1}{2} \sum_{i \in \mathrm{creases}} \kappa_i (\rho_{\mathrm{D}})_i^2 \tag{4.19}$$

This function is manifestly linear in κ_i, such that minimizing it (subject to the linear constraints) can be easily performed performed using Linear Programming, an efficient polynomial algorithm. As the linear optimization function generically looks for solution with small κ values, we find that this algorithm obtains solutions that saturate the lifting constraint of Equation 4.2.

Geometric distortion involves changing the minimal energy configurations due to the crease stiffness. We find this kind of distortion is present in every case that the creases of an origami pattern are made stiffer in a heterogeneous manner. If one wishes to design a pattern to have a specific precise folded state, it might be important to control geometric distortions due to stiff creases. To reduce the geometric distortion in a folded state we can optimize the energy due to stiff creases such that its angular minimum (energy minimum at given ρ) is close to the desired folded state. The gradient of crease energy at the desired mode configuration is:

$$\frac{\partial E}{\partial \rho_i}(\boldsymbol{\rho} = \boldsymbol{\rho}_{\mathrm{D}}) = (\boldsymbol{\kappa} \star \boldsymbol{\rho}_{\mathrm{D}})_i, \tag{4.20}$$

where $\star$ indicates element wise multiplication. If ρ_{D} is an exact angular minimum of the energy function, we know that the normalized dot product between $\frac{\partial E}{\partial \rho_i}$ and $(\rho_{\mathrm{D}})_i$ has to be unity, as the angular components of the gradient vanish and the force is entirely in the ρ_{D} direction:

$$\frac{\frac{\partial E}{\partial \rho_i}(\boldsymbol{\rho} = \boldsymbol{\rho}_{\mathrm{D}}) \cdot (\rho_{\mathrm{D}})_i}{\|\frac{\partial E}{\partial \rho_i}(\boldsymbol{\rho} = \boldsymbol{\rho}_{\mathrm{D}})\| \cdot \|(\rho_{\mathrm{D}})_i\|} = 1, \tag{4.21}$$

where the numerator multiplies the vectors ρ_{D} and the energy gradient at $\boldsymbol{\rho} = \boldsymbol{\rho}_{\mathrm{D}}$, while the denominator multiplies their norms. By writing these terms explicitly and rearranging them, we find that a configuration ρ_D that has no geometric distortion satisfies

$$\rho_{\mathrm{D}}^2 \sum_{i \in \text{creases}} \kappa_i^2 (\rho_{\mathrm{D}})_i^2 \; - \sum_{i,j \in \text{creases}} \kappa_i \kappa_j (\rho_{\mathrm{D}})_i^2 (\rho_{\mathrm{D}})_j^2 = 0 \tag{4.22}$$

Evidently, a solution for which this condition is satisfied is $\kappa_i = \text{const}$. Unfortunately, a homogeneous crease stiffness profile does not lift any of the modes near the flat state. We would like to find stiffness profiles that are heterogeneous to lift all but one minimum, but still optimize geometric distortion. This can be done if instead of demanding the condition of Supplementary Equation 4.22 be satisfied, we treat it as a function to be minimized subject to the linear constraints:

$$F_{\mathrm{QP}} \equiv \rho_{\mathrm{D}}^2 \sum_{i \in \text{creases}} \kappa_i^2 (\rho_{\mathrm{D}})_i^2 \; - \sum_{i,j \in \text{creases}} \kappa_i \kappa_j (\rho_{\mathrm{D}})_i^2 (\rho_{\mathrm{D}})_j^2 \tag{4.23}$$

Crucially, the function F_{QP} is manifestly quadratic in the stiffness profile κ_i, and positive semi-definite. These properties allow us to optimize F_{QP} with Quadratic Programming, another efficient polynomial algorithm. We use MATLAB Linear and Quadratic Programming routines to find these stiffness profile solutions. The main text shows how these schemes optimize energetic and geometric distortions of the chosen vertex motion.

For single 4-vertices, one could sample random stiffness profiles that satisfy the linear constraints and come close to optimizing mode distortions. The major gains of using our stiffness optimization protocols manifest when considering larger patterns. As shown in Supplementary Figure 4.9(a), even for simple looped patterns (quads), LP and QP stiffness protocols achieve optimized distortion values better by orders of magnitude compared to other schemes, including the $\boldsymbol{\rho}_{\mathrm{D}}^{-n}$ prescription that worked well for single vertices. The superior performance of the optimized protocols was established over a sample of 100 patterns.

For larger patterns still, random stiffness profiles that satisfy the linear constraints are usually accompanied by very high energetic and geometric distortions. Our optimized stiffness selection

protocols are superior to any other choice, yet are themselves subject to degradation for large patterns. Supplementary Figure 4.9(b-c), shows how the optimum energetic and geometric distortions (achieved by LP and QP, respectively) grow with pattern size.

Finally, we tested the robustness of our method to manufacturing errors in the stiffness profile. We simulated such errors in small and large patterns by perturbing the optimal solutions (obtained by either LP or QP) with relative values of up to 5%. We find that the resulting unique low-ρ mode is essentially a perturbed version of the one obtained for the optimal solution. The fact that small perturbations in κ lead to small distortion of the unique low-ρ mode means that our method is expected to be robust to manufacturing errors in the stiffness profile κ.

Supplementary Note 5 - Adiabatic failures and bifurcations along the desired pathway

The main text discusses how selecting an appropriate crease stiffness profile establishes a smooth folding motion connecting a unique minimum close to the flat state and a desired nonlinear folded configuration. Indeed, we find that the designed folded configuration frequently connects continuously to the unique small ρ minimum established by stiffness profiles due to LP or QP springs. This gives rise to the heuristic approach of folding self-folding origami: Design the pattern with a heterogeneous crease stiffness profile (solving LP or QP), and fold the pattern slowly.

Although this idea appears to work for many crease patterns ($> 90\%$ of the 3×3 patterns in our sample), it is not universally successful. In some patterns (Supplementary Figure 4.10) we observe that the desired folded configuration does not connect continuously to the unique low-ρ minimum, but instead terminates at a saddle-node bifurcation. In turn, the unique low-ρ minimum itself is also terminated with a saddle node-bifurcation. Thus, failure to fold adiabatically is associated with a saddle-node bifurcation along the desired pathway.

Such failures can be mathematically ruled out if the states along adiabatic pathway are always the lowest energy state for each ρ since ground states cannot undergo saddle-node bifurcations.

In fact, we know that the pathway has lowest energy at low and high ρ — there is only one state at low ρ and at high, the pathway connects with the desired mode which is designed to be the lowest energy state by solving loop equations [21]. However, such protection does not extend to intermediate ρ. If a lower energy state exists at finite ρ, the desired pathway could undergo a saddle-node bifurcation like that shown in Supplementary Figure 4.10, preventing adiabatic folding.

In the particular case presented here, the likely candidate is a distractor configuration shown in blue. When this distractor folded state is unfolded slowly, it morphs into a state that has smaller energy than the unique low-ρ minimum. This allows the unique low-ρ minimum to terminate in a saddle-node bifurcation. When folded slowly from the flat state, the adiabatic pathway described by the unique low-ρ minimum abruptly terminates, and the folded state snaps into another configuration, which itself adiabatically morphs into a distractor mode.

Chapter 5

Supervised learning in a mechanical system

The design of mechanical metamaterials usually assumes that desired force-response properties are given as a top-down specification. For example, principles of topological protection can be used to design materials where forces at specific sites lead to localized deformations [17, 112]. Elastic networks can be pruned to exhibit allostery [15], so that a deformation at one specific site is communicated to a specific distant site. In these examples and many others [1, 21, 23, 113, 114], we rationally optimize design parameters, e.g., spring constants and geometry, to achieve a specified force-response relationship.

A different scenario, closely connected to supervised learning in computer science, is when the desired force-response is so complex that it cannot be specified in a top-down manner; however, it might still be easy to give examples of the desired force-response relationship. The goal is to learn or infer the right force-response relationship from these *training* examples, with success evaluated on the ability to extrapolate the learned relationship to unseen *test* examples. Learning from examples in this manner offers several advantages for materials, primarily in the form of tailoring the force-response relationship to real use cases. Consider a class of spatial force patterns (e.g. exerted by cat paws), which when applied to a sheet, should fold it into one geometry and another class of force patterns (dog paws) that should result in a different folded geometry. While it might be easy to obtain and apply physical examples of forces from these classes, it is hard to

mathematically list what features distinguish these two classes, especially given the large variation within the classes themselves. A learning process that automatically learns the right features from a training set can naturally solve this problem. Second, even when distinguishing features are known, learning offers a natural way of arriving at the right design parameters without need for a complex optimization algorithm. Finally, and most critically, successful learning promises a material that can show the correct response to novel inputs not seen during training.

While naturally occurring systems like neural networks [115], slime molds [116], and plant transport networks [117] use similar ideas to adapt their response to environmental inputs, physical supervised learning has thus far not been used to obtain functional man-made materials. Here we propose an approach for the supervised training of a mechanical material through repeated physical actuation. We work with a model of creased thin sheets where crease stiffnesses can change as a result of repeated folding. We assume a training set, that is, a list of force patterns and desired responses. Each training example of force pattern is applied to the sheet; if the response is the desired one, as determined by a 'supervisor', folded creases are allowed to soften in proportion to their folding strain. If the response is incorrect, creases stiffen instead. We then test the trained sheet by applying unseen force patterns (test examples) drawn from the same underlying distribution as the training data. We study test and training errors and thus the sheet's ability to generalize to novel patterns as a function of its size.

Our proposal here relies on a plastic element, namely crease stiffness. Materials that stiffen or soften with strain have been demonstrated in several contexts [118, 119, 120, 121], including recently in the training of mechanical metamaterials [8]. We discuss how learning performance may be affected by limitations on the dynamic range of stiffness and other practical constraints in such materials. We hope our results here will provoke further work on how the constraints of mechanics intersect with learning.

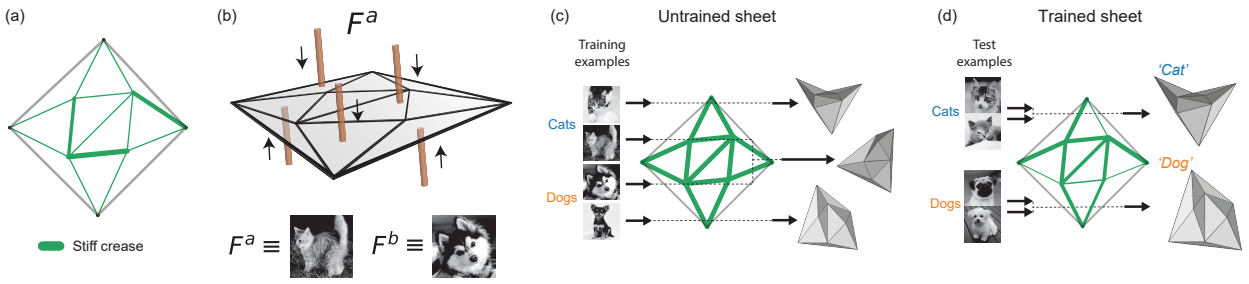

Figure 5.1: Training thin sheets to classify spatial force patterns
a) We consider thin stiff sheets with creases whose stiffnesses (indicated by thickness of green segments) can be changed by repeated folding. b) Such sheets can fold in response to a spatial force pattern $\mathbf{F^a}$ applied across the sheet. To emphasize the high dimensional nature of $\mathbf{F^a}$, we visualize $\mathbf{F^a}$ as an image where the grayscale value of each pixel corresponds to the force at a particular location on the sheet. c) An untrained sheet with uniform stiffness shows random folded responses for different spatial force patterns. d) By modifying crease stiffness values, we train the sheet to classify entire classes of force patterns by showing one distinct folded response for each class.

5.1 Results

We demonstrate our results with a creased thin self-folding sheet (Fig. 5.1a), which is naturally multi-stable. Our analysis can be generalized to other disordered mechanical systems, such as elastic networks [8], that are also generically multi-stable. It was previously shown that creased sheets, such as those of self-folding origami, can be folded into exponentially many discrete folded structures from the flat, unfolded state [22, 81]. Such exponential multi-stability can be a problem [22, 23] from an engineering standpoint, as precise controlled folding is required to obtain the desired folded structure.

Here we exploit such multi-stability to train a classifier of input force patterns. If we apply a spatial force pattern $\boldsymbol{F}$ across the flat sheet (see Fig. 5.1a-b), the sheet will fold into a particular folded structure $\boldsymbol{\rho}(\boldsymbol{F})$, e.g., described by dihedral folding angles at each crease ρ_i (see Supplementary Note 1). To emphasize the high dimensional nature of space of force patterns, in Fig. 5.1b-d, we represent each force with a gray-scale image where each pixel can be interpreted as a force at a designated point on the sheet. The set of all force patterns $\{\boldsymbol{F}\}$ that lead to one particular folded

structure $\boldsymbol{\rho}^m$ is defined as the 'attractor' of folded structure m in the space of force patterns (color coded regions in Fig. 5.2b). The complex attractor structure of force-response for a thin sheet naturally serves as a classifier of force patterns, albeit a random classifier (Fig. 5.1c). The goal of the training protocol is obtain a sheet with a specific desired mapping between force patterns and folded structures (Fig. 5.1d).

Previously, we found that the folded response to a given force pattern can be modified by changing the stiffness k_i of different creases i in the sheet [23]. Here, we employ a 'supervised learning' approach to naturally tune stiffness values k_i so that the sheet classifies forces as desired. Intuitively, this is done by applying examples of force patterns to the sheet and modifying crease stiffness accordingly, in a way that reinforces the correct response and discourages incorrect folding (Fig. 5.2a). Such training, carried out iteratively for different force pattern examples, has the effect of morphing the attractor structure to better approximate the desired response (Fig.5.2b).

Consider two distributions of force patterns, each designated as a particular class (e.g. 'cats' and 'dogs'). An example is shown in Fig. 5.3a (top) where the two classes of forces are defined by spherical caps in force space. We define all forces belonging to these distributions by $S^{\text{dog}} = \{\boldsymbol{F} | \boldsymbol{F} \cdot \boldsymbol{F}_{\text{dog}} \geq D, \boldsymbol{F} \cdot \boldsymbol{F}_{\text{dog}} > \boldsymbol{F} \cdot \boldsymbol{F}_{\text{cat}}\}$, and similarly for S^{cat}. Here $D = 0.6$ sets the size of the caps. (In Fig. 5.3a, S^{dog} is blue and S^{cat} is orange.)

Assume we are given two sets of labeled force patterns as training examples $\mathcal{F}^{\text{dog}} = \{\boldsymbol{F} \in S^{\text{dog}}\}$, $\mathcal{F}^{\text{cat}} = \{\boldsymbol{F} \in S^{\text{cat}}\}$, each with n training force patterns (in Fig. 5.3a (bottom) we sample sets with $n = 20$). Together, $\mathcal{F}^{\text{dog}}$ and $\mathcal{F}^{\text{cat}}$ are defined as the *training set*. We desire all forces in S^{dog} to result in one common folded structure, while all forces in S^{cat} fold the sheet to a distinct but common folded structure. While S^{dog}, S^{cat} are separable in some $2d$ projection of force space, learning is non-trivial since the sheet must learn the 2 dimensions in which these distributions are separable.

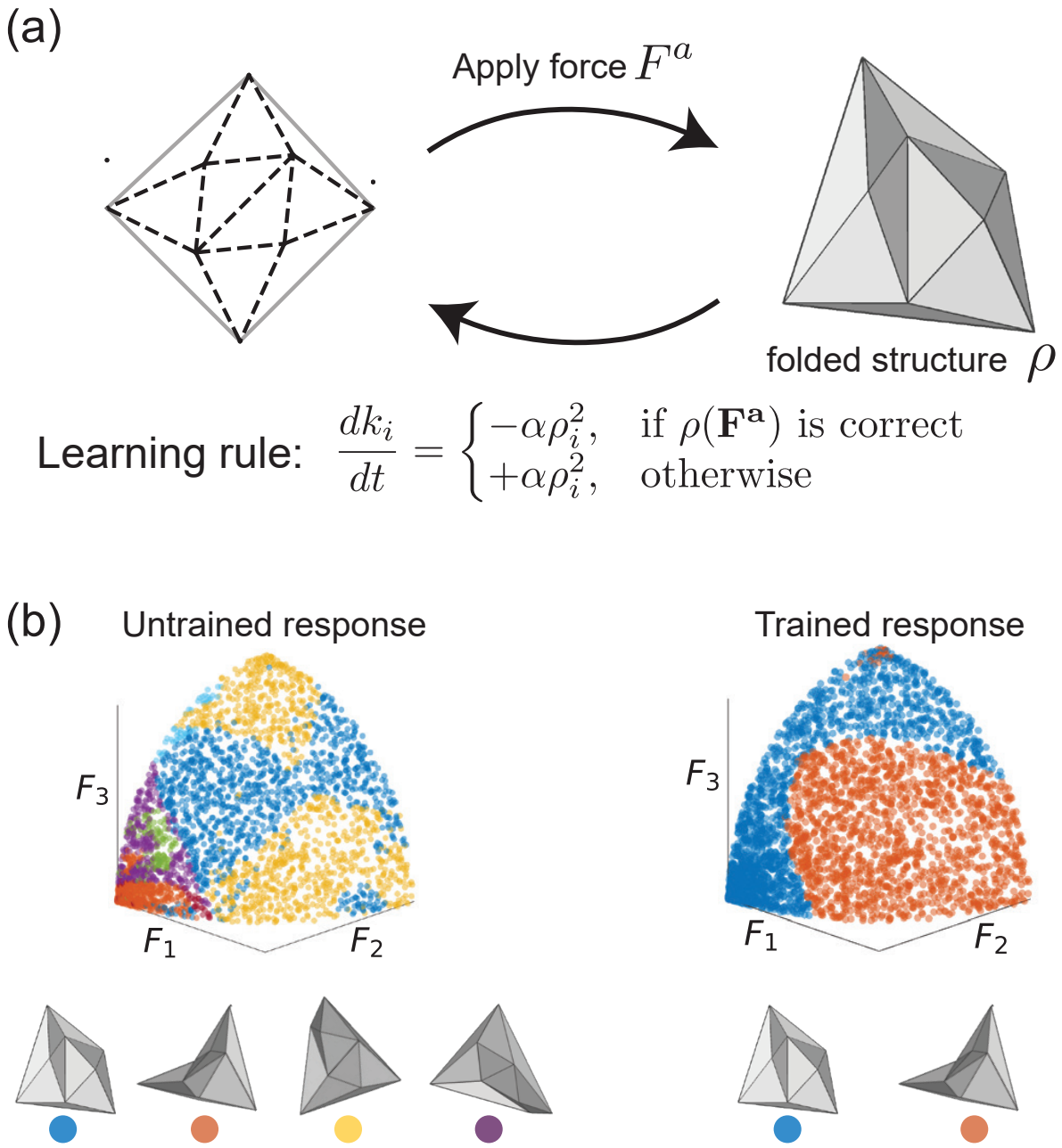

Figure 5.2: Supervised training of thin sheets

a) A sheet with random crease geometry is folded with a training force pattern $\boldsymbol{F}^a$, resulting in a folded structure $\boldsymbol{\rho}$. The stiffness k_i of each crease i is modified according to a local learning rule; if the folded structure $\boldsymbol{\rho}$ is the desired response for $\boldsymbol{F}^a$ as determined by a supervisor, creases soften in proportion to their folding strain ρ_i. Otherwise, creases stiffen. b) This rule trains the sheet to perform the desired classification of force patterns. The untrained sheet shows multiple folded structures in response to force patterns ($2d$ cross-section of force pattern space shown). The trained sheet shows only two folded responses that mimic the desired mapping of force patterns to folded structures.

5.1.1 A mechanical supervised training protocol

In our training protocol, each of the training force patterns $\boldsymbol{F}^a$ is applied to the sheet in sequence, to obtain a folded structure $\boldsymbol{\rho}(\boldsymbol{F}^a)$. A supervisor determines whether the resulting folded state $\boldsymbol{\rho}(\boldsymbol{F}^a)$ is correct or incorrect by comparing it to a reference state $\boldsymbol{\rho}_{\text{ref}}(\boldsymbol{F})$ for those classes. (The reference state can be selected in several ways. Here, we average the response of the untrained sheet on training examples in each class; see Supplementary Note 2.)

We then apply the following learning rule that stiffens or softens each crease in proportion to folding in that crease,

$$\frac{dk_i}{dt} = \begin{cases} -\alpha\rho_i^r, & \text{if } \boldsymbol{\rho}(\boldsymbol{F}^a) \text{ is correct} \\ +\alpha\rho_i^r, & \text{otherwise} \end{cases} \tag{5.1}$$

for the stiffness k_i of each crease i. α is a *learning rate*, setting how fast stiffness values k_i are updated due to training examples. r models non-linearities in strain-based softening or stiffening of materials; we use $r = 2$. Such plasticity is experimentally seen in several materials [122, 123, 124]; we discuss other learning rules and experimental constraints later.

After each round of training the pattern is unfolded back to the flat state. The same supervised learning step is then repeated in sequence for all training force patterns. A training epoch is defined as one pass through the entire training set.

We find that as training proceeds, the number of observed folded structures decreases (fewer colors), and nearly all training force patterns fold the sheet into the 'blue' or the 'orange' labeled structures after epoch 40 (diamonds in Fig. 5.3b). The fraction of training force patterns that fold the sheet into the correct structure is defined as the *training accuracy*.

However, a successfully trained sheet should correctly classify previously unseen 'test' force patterns, sampled from the same distributions. We tested the trained sheet by applying such test examples drawn from the caps $S^{\text{dog}}, S^{\text{cat}}$ and recording the resulting folded structure. In analogy to the training sets, the fraction of test examples yielding the correct folded structure is defined as the *test accuracy*. High test accuracy is observed (Fig. 5.3c,d) ($\sim$ 80% of the test examples

classified correctly); thus the sheet generalizes and is able to have the right response to novel test force patterns through the changes induced by training examples.

5.1.2 Heterogeneous crease stiffness

Our learning rule facilitates classification by creating heterogeneous crease stiffness across the sheet (Fig. 5.4a). Indeed, as training proceeds, we find that the variance Δk^2 of stiffness grows (Fig. 5.4b). If the sheet is trained beyond the optimal point, the stiffness variance still grows, but the classifier eventually fails, as seen in Fig. 5.3b-c. The failure mode of over-training is typically that all forces fold the sheet into a single folded structure, resulting in no classification.

We can understand this relationship between heterogeneous stiffness (Δk) and training using a simple model. A heterogeneous crease stiffness profile $\boldsymbol{k}$ with high stiffness k_i in crease i but no stiffness elsewhere, will lift the energy of structures $\boldsymbol{\rho}$ with small folding ρ_i in crease i less than structures with large ρ_i. Hence heterogeneous $\boldsymbol{k}$ can raise the energy of select structures, reducing their attractor size, while other structures remain low in energy and grow in attractor size. If we assume that folding angles $\boldsymbol{\rho}^a$ of structure a are randomly distributed (verified earlier in [21]) and assume a random stiffness pattern with standard deviation Δk, the energies $\sum_i k_i \rho_i^2$ of different structures will be distributed as a Gaussian with mean $\mu = \alpha \bar{k}$ and standard deviation $\sigma = \beta \Delta k$ where $\bar{k}$ is the mean stiffness, and α, β some numerical parameters.

If structures above energy E_F are inaccessible to folding, the number of accessible folded structures is,

$$\#(\Delta k, N) \sim 2^N [1 - \mathrm{erf}(\frac{\alpha \bar{k} - E_F}{\beta \Delta k})]. \tag{5.2}$$

Hence the number of surviving folded structures should decrease fast with Δk. This effect is indeed observed for trained origami sheets of different sizes (Fig. 5.4d). From numerical exploration of the energy distributions in this model, we find that α is a constant regardless of sheet size, while $\beta \sim N^{-0.5}$ shrinks with sheet size (central limit theorem). Using this form of β in Eq. 5.2 predicts that the elimination of structures happens at a lower Δk for larger sheets, consistent with

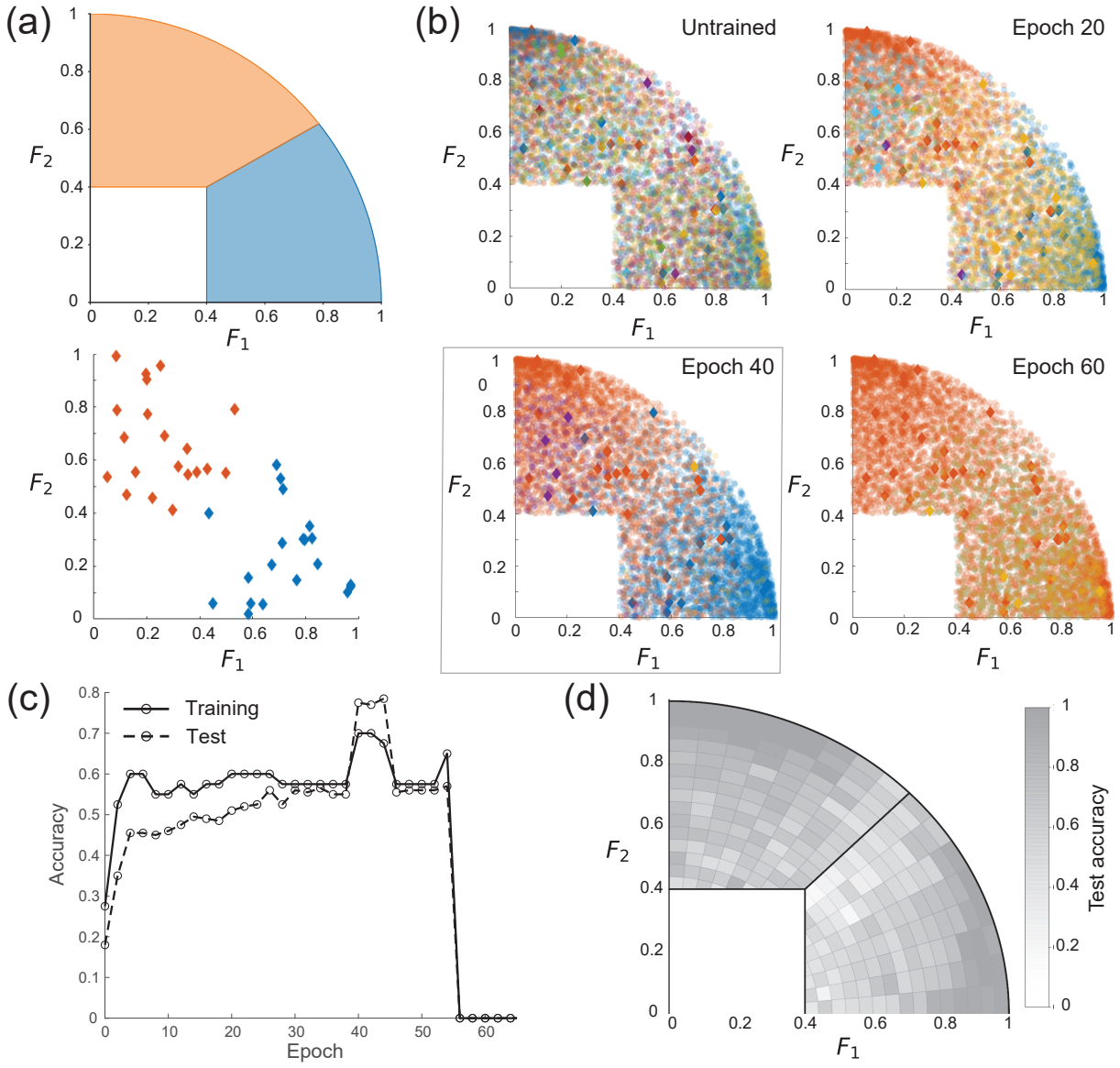

Figure 5.3: Supervised learning of cap-like force distributions
a) We define distributions S^{dog} (blue) and S^{dog} (orange) of force patterns as two spherical caps in the space of applied forces (2d projection shown). Training examples (diamonds) are drawn from both distributions. b) An untrained sheet folds into many distinct folded structures (different colors) in response to applied force patterns. As training progresses, most force patterns are classified as either blue or orange according to the cap they belong to. When over-trained, all force patterns result in only one folded structure (orange). c) The trained sheet reaches peak performance after $\sim$ 40 epochs of supervised training (i.e., passes through the training examples). The trained sheet not only classifies the training set correctly (training accuracy), but generalizes to unseen test force patterns (test accuracy). d) The trained sheet is highly accurate when classifying force patterns near the center of the spherical caps, but less accurate close to the true decision boundary between the distributions.

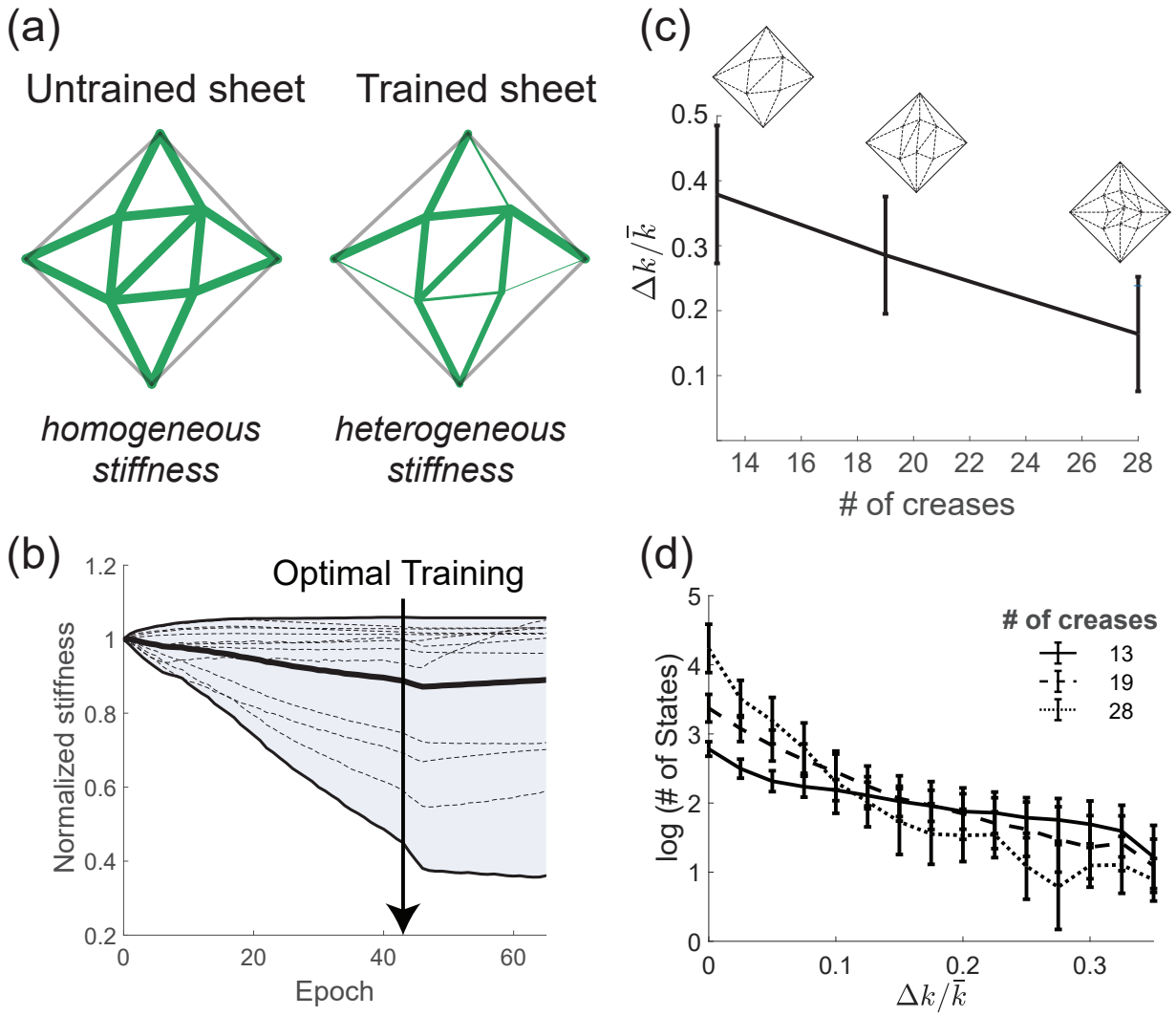

Figure 5.4: Training increases the variance of crease stiffness across the sheet
a) Untrained sheets have a homogeneous distribution of crease stiffnesses, while trained sheets have heterogeneous stiffness profiles (width of green lines). b) As the sheet is trained, the stiffness of different creases changes to different extents, such that the variance in stiffness values grows over training time (envelope shows the least and most stiff creases). c) Larger sheets with more creases require smaller variance in their stiffness values for optimal training. d) An untrained sheet starts with exponentially many available folded structures. During training, the number of available folded structures decreases exponentially with increasing stiffness variance Δk^2, allowing the sheet to classify a few distinct classes.

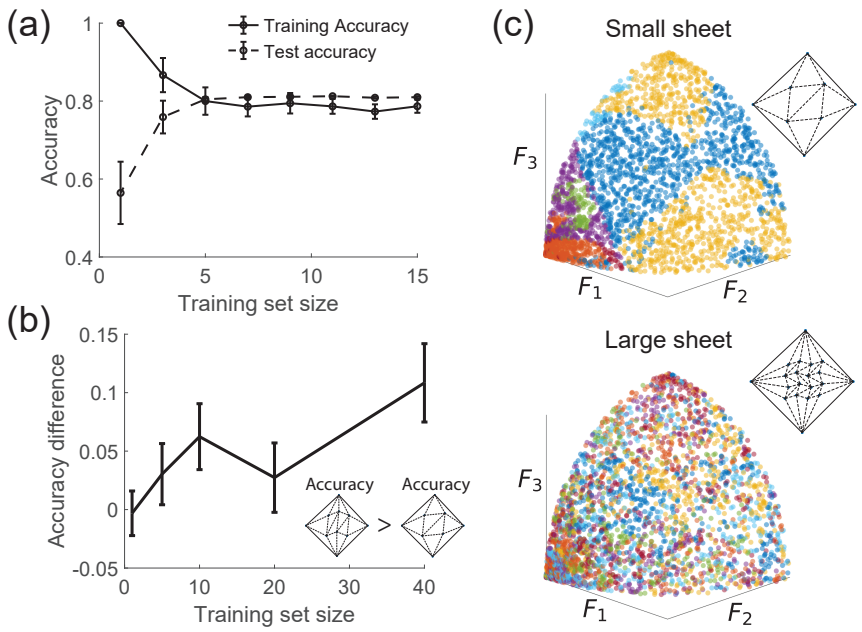

Figure 5.5: Effect of training set size and sheet size on test accuracy
a) With fewer distinct training examples, training accuracy is high but test accuracy is low (overfitting). Increasing the number of training examples improves test accuracy, at the expense of training accuracy. b) Sheets with more creases show larger improvements in test accuracy with increasing number training examples, as expected of complex models with more fitting parameters. c) A small untrained sheet (13 creases) shows ~ 10 folded structures (color coded) in response to different force patterns. A larger sheet (49 creases) sheet shows ~ 400 folded structures instead, each with smaller attractor regions in the space of force patterns. Consequently, larger sheets can create more flexible classification surfaces by combining smaller attractor regions; such complex models with more fitting parameters require more training examples to avoid overfitting.

our results in Fig. 5.4c-d.

We conclude that as the training protocol proceeds, the stiffness variance Δk^2 grows, and the number of available folded structures decreases. The last surviving folded structures, reinforced by the learning rule of Eq (5.1), classify the force distributions correctly. Thus, the learning process merges attractors of the untrained sheet such that the surviving attractors recapitulate features of the desired force-fold mapping.

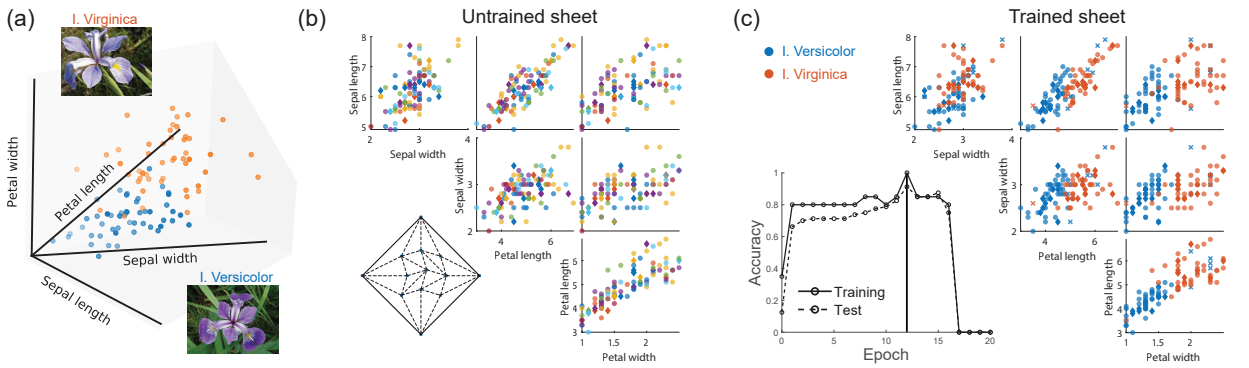

Figure 5.6: Training sheets to classify Iris specimens
a) We train a sheet to classify individual Iris specimens as one of two species based on petal and sepal lengths and widths [125]. We translate these four measurements into a spatial pattern of forces applied to the sheet. b) Folding response of an untrained (28 crease) sheet due to force patterns derived from the Iris data (shown in every cross section). c) The sheet is trained using 10 random examples (diamonds) of each species from the database [125] and then tested on 80 unseen test examples (circles). Matrix shows the classification of Iris flowers at optimal training of the sheet (91% test accuracy; mistakes denoted by x).

5.1.3 Generalization and sheet size

Statistical learning theory [126] suggests that two critical parameters set the quality of learning: (1) the number of training examples seen, (2) complexity of the learning model. An increased number of training examples usually decreases training accuracy. However, test accuracy - i.e., the response to novel examples or the ability to generalize - improves. Furthermore, the improvement of test accuracy is larger for complex models with more fitting parameters. Intuitively, complex models 'overfit' details of small training sets, and thus show low test accuracy even if training accuracy is high.

Our sheets exhibit signatures of these learning theory results, with the size of the sheet (number of creases) playing the role of model complexity. For a sheet of fixed size, trained on the distributions of Fig. 5.3, we observe that increasing the number of training examples increases test accuracy and decreases training accuracy (Fig. 5.5a). In Fig. 5.5b, we find the test accuracy of larger sheets with more creases improves more dramatically with the size of the training set, compared to smaller sheets.

These results suggest that sheets with more creases correspond to more complex classification models (e.g., a neural network with more neurons). For example, crease stiffnesses are the learnable parameters in our approach; hence increasing their number amounts to using a training model with more parameters. Further, untrained sheets with more creases support exponentially more folded structures [22, 81] as shown by the color coded force-to-folded-structure relationship in Fig. 5.5c. The training protocol achieves correct classification by merging different colored regions. Thus, larger sheets can approximate more complex decision boundaries by combining the smaller regions shown in Fig. 5.5c., and thus act as more complex models to be favored when the amount of training data is large. In the Discussions, we use these results to contrast memory and learning in mechanical systems.

5.1.4 Complex classification problems

The attractor structure of disordered thin sheets is complex, and contains an exponential number of attractors. Hence, sheets can be expected to learn more complex features than those shown previously in Fig. 5.3.

We tested our learning protocol on the classic Iris data set [127], used extensively in the past to benchmark classification algorithms. This data set reports four measurements - length and width of petals and sepals - for individual specimens of different Iris species. While different Iris species cannot be distinguished by any one of these properties, we wanted to test if our sheet can learn the combination of features needed to distinguish species.

We picked the two most similar species in this data set, *Iris Versicolor* and *Iris Virginica* (Fig. 5.6a). We translated the four flower measurements to four force components applied to a sheet (see Supplementary Note 4). We then applied our training algorithm with a training set consisting of 10 examples of *I. Versicolor* and *I. Virginica* (diamonds in Fig. 5.6c). The resulting trained sheet was tested on 80 *unseen* examples of these species; the trained sheet was able to identify the species of 91% of previously unseen specimens correctly.

We have tested our training protocol on more complex, higher dimensional distributions (Sup-

plementary Note 3). For example, we used the folding behavior of one thin sheet (the master) as the target behavior for another thin sheet with a distinct crease geometry. We find that the trained sheet is able to correctly predict the response of the master sheet to forces not seen during training. Thus, using our training protocol, sheets can learn and generalize complex force-to-folded-response maps from examples.

5.1.5 Experimental considerations

Our learning framework requires materials that can plastically stiffen or soften when strained repeatedly [128]. Several such materials and structures are known, including shape memory polymers [129, 130], shape memory alloys (Nitinol) [131], and fluidic flexible matrix composites [132]. Such materials have the advantage of truly variable, user controlled stiffness. Other materials can show a plastic change in stiffness in response to aging under strain, such as Ethylene Vinyl Acetate (EVA) foam [133] and thermoplastic Polyurethane [134]. EVA was used recently [8] to show such behavior in a mechanical system trained for auxetic response.

The specific learning rule used in the paper requires the ability to soften or stiffen depending on the supervisor's judgement of outcome. Such a learning rule can be implemented by materials that stiffen under strain in one condition (say, high temperature, low pH) but soften under strain in another condition.

However, the results here also hold for simpler learning rules, e.g., that only require plastic softening under strain. For example, we can modify the learning rule (Eq. 5.1) to:

$$\frac{dk_i}{dt} = \begin{cases} -\alpha\rho_i^2, & \text{if } \boldsymbol{\rho}(\boldsymbol{F}) \text{ is correct} \\ 0, & \text{otherwise} \end{cases} \tag{5.3}$$

Such a rule is easily implemented with a strain-softening material with no stiffening needed. For example, if the folded outcome $\boldsymbol{\rho}(\boldsymbol{F})$ is judged correct, we hold the sheet in the folded state $\boldsymbol{\rho}(\boldsymbol{F})$ for a longer length of time (allowing significant softening) than when the outcome is judged incorrect (no softening). We tested this simplified learning rule for the classification problem in

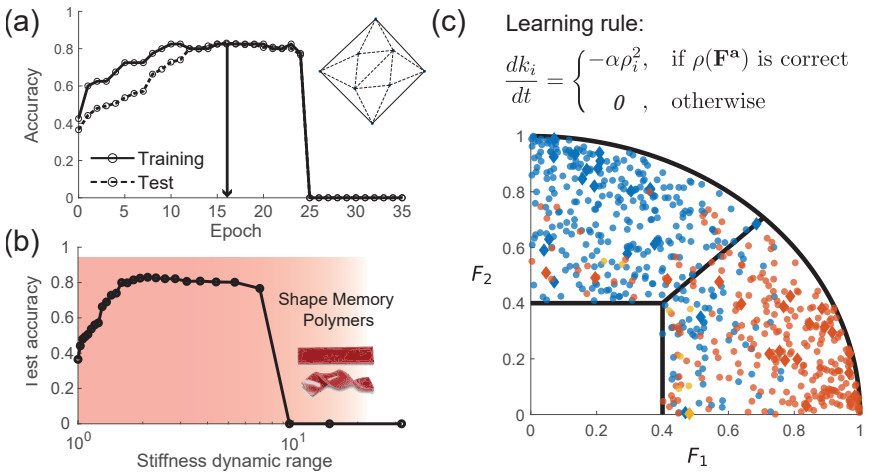

Figure 5.7: Learning is successful even with simplified training rules and experimentally realizable stiffness range
a) A sheet (13 crease) trained on the classification problem of Fig. 5.3, with a simplified, experimentally viable learning rule shown in (c). b) At peak training, the dynamic range of crease stiffness values is ~ 2, well within the ranges supported by existing shape memory polymers (red filled region) [129]. c) Trained sheet reaches a peak accuracy of $\sim 80\%$ on test force examples (circles).

Fig. 5.3; we find similar accuracy as earlier (see Fig. 5.7a).

Another significant experimental constraint is the dynamic range of crease stiffnesses k_i achievable in real materials without failure or fracture at the creases. Fortunately, we find that for well trained sheets, the difference in crease stiffness is moderate (Fig. 5.4c), and does not exceed 30% of the mean stiffness value for a medium sized (28 crease) sheet. Fig. 5.7b shows that our required dynamic range in stiffness is within the range for experimentally available materials such as shape memory polymers [129, 135].

Finally, another failure mode for our training protocol is overtraining. While the variance in crease stiffness Δk^2 is critical to eliminate attractors, overtraining can result in a sheet with only one folded structure. Our analysis, presented in Fig. 5.4, suggests that large sheets should be easier to train experimentally since the stiffness variance needed is more moderate, while the transition to overtraining does not become much more rapid.

5.2 Discussion

In this work, we have demonstrated the supervised training of a mechanical system, a thin creased sheet, to classify input force patterns. As required for learning, the trained sheet not only shows the correct response for training forces, but can generalize and show the correct response to unseen test examples of forces. We studied the relationship between training error, test error, and the size of the sheet which plays the role of model complexity in supervised learning [126].

We can contrast the learning framework here with that of memory formation in mechanical systems [136]. A robust memory implies a trained model that shows the correct response for all of the training examples (i.e., low training error), even at the expense of overfitting such training data. In contrast, in learning, we seek a model that generalizes and responds correctly to unseen examples (i.e., low test error), even at some expense of training error. With this view, large sheets trained with a small number of training examples can serve as memory while smaller sheets with more training examples lead to generalization, and hence learning.

Supervised training of mechanical systems offers advantages over traditional mechanical design. On a practical level, a material with an intrinsic mechanical learning rule can be trained by an end-user rather than an expert designer, according to the task at hand. In the case of sheets presented here, the same sheet can be used to classify different data distributions depending on how it was trained. Such properties are highly sought after, e.g. in adaptive robotics [137]. Finally, as learning allows generalization, materials can be trained to show a desired force-response behavior even if some examples of use are not available at the time of training.

5.3 Supplementary Notes

Supplementary Note 1 - Folding origami sheets

Energy of folded structures

The origami sheets used in this work are based on a self-folding origami energy model developed and validated in previous studies [21, 37, 42]. The effects of stiff creases are modeled by using torsional spring elements on each crease [22, 23]. Here we discuss in detail how the energy of a folded structure is computed.

For thin origami sheets with free-folding creases, the primary contribution to the energy of a folded structure is due to bending of the sheet faces. Instead of modelling the faces directly, we look at the mechanical constraints inherent to the geometry of the vertices. An origami vertex is known to apply 3 constraints on the dihedral folding angles of the creases connected to it (due to to embedding of the sheet in $3d$-space). The constraints can be derived by noting that the vertex must not tear open when folded. Thus, starting from any crease, alternating rotations about the dihedral and sector angles around the vertex have to result in an identity operation [22, 23, 37].

Suppose there are N creases denoted by an index i, each forded to an angle ρ_i, and N sectors with angles θ_i around the vertex. Rotations about one dihedral angle and one sector would combine to form a rotation matrix

$$R_i = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\rho_i & -\sin\rho_i \\ 0 & \sin\rho_i & \cos\rho_i \end{pmatrix} \begin{pmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{5.4}$$

For the vertex to be closed (i.e. not torn open) in a folded structure, the combination of rotation about all crease dihedral angles and sector angles must be the identity:

$$A \equiv \prod_{i=1}^{n} R_i = I. \tag{5.5}$$

A folded structure with values ρ_i that do not satisfy Eq. 5.5 must cause the sheet faces to bend. Mathematically, this effect will manifest in finite off-diagonal values in the matrix $A \equiv \prod_{i=1}^{N} R_i$. As there are 3 independent non-diagonal elements, we say that the vertex imparts 3 mechanical constrains on the dihedral angles ρ_i around it.

At the flat state all $\rho_i = 0$ all constraints are trivially satisfied, so we can write down an expansion for the 3 off-diagonal terms of A ($T_1 \equiv A_{12}, T_2 \equiv A_{13}, T_3 \equiv A_{23}$) in powers of the folding angles:

$$T_a(\rho_i) = C_a^i \rho_i + D_a^{ij} \rho_i \rho_j + \dots \tag{5.6}$$

Then, the energy of breaking these constraints is taken as the sum of squares of the residues T_a of the constraint equations $E_{\text{Vertex}} \sim \sum_a T_a(\rho_i)^2$. Summing this vertex energy over all the vertices of the sheet gives rise to the total face bending energy. The energy due to folding of a stiff crease (modeled as a torsional spring with modulus κ_i) is quadratic in the folding angle $E_{\text{Crease},i} = \frac{1}{2} k_i \rho_i^2$. The total energy of a folded sheet with stiff creases is thus computed as

$$E_{\text{sheet}}(\boldsymbol{\rho}) \equiv E_{\text{Face}} + E_{\text{Crease}} = \sum_{v \in vertices} \sum_{a=1}^{3} T_{va}(\boldsymbol{\rho}_v)^2 + \frac{1}{2} \sum_{i \in \text{creases}} k_i \rho_i^2. \tag{5.7}$$

Note that we do not assign an explicit energy scale to the face bending term, so it is implicitly combined with the crease stiffness scale $\bar{k}$.

Folding protocol

Now that the energy of every folded structure ρ_i of a specific sheet is defined. We can use this energy landscape to simulate the folding of the sheet. Experimentally there are multiple different ways to fold origami sheets [10], and we have previously outlined how these methods can be

simulated numerically [23].

One way that an origami sheet can be folded is by applying torques directly to the different creases. Suppose a crease i of a flat sheet is subjected to an external torque F_i^{ext}. Such a torque will induce folding in the crease, but the sheet generally resists folding due to the extra energy that might be associated with a folded structure. Assuming that the folding process is over-damped, we may write a dynamical folding equation

$$\tau_{\text{relax}} \frac{d\rho_i}{dt} = -\frac{\partial E_{\text{sheet}}(\boldsymbol{\rho})}{\partial \rho_i} + F_i^{\text{ext}}, \tag{5.8}$$

where $\boldsymbol{\rho}$ is the current folded structure, and τ_{relax} a time scale of the over-damped dynamics. In this work we utilize a specific way of folding the origami sheets. Suppose a set of external torques $\boldsymbol{F}^{ext}$ is given (this could be a training or a test example as described in the main text). First, the sheet is folded very fast with a strong external torque $\boldsymbol{F}^{ext}$, until a certain folding magnitude $\rho \equiv \|\boldsymbol{\rho}\|$ is reached. For fast folding we can initially disregard the sheet energy and thus get to a state

$$\boldsymbol{\rho}_{\text{fast}} = \rho \frac{\boldsymbol{F}^{ext}}{\|\boldsymbol{F}^{ext}\|}.$$

Then the sheet is relaxed subject to the constraint that the overall folding magnitude is fixed (i.e. finding an energy minimum on a hyper-sphere of radius ρ in ρ-space):

$$\begin{aligned} &\underset{\rho_i}{\text{minimize}} \quad E_{\text{sheet}}(\boldsymbol{\rho}) \\ &\text{subject to} \quad \|\boldsymbol{\rho}\| = \rho. \end{aligned} \tag{5.9}$$

Finding a local minimum on the hyper-sphere guarantees that this folded structure would naturally occur if the sheet is folded with appropriate torques, as any neighboring configuration costs more energy, and the local minimum will attract the folding process. This algorithm is used to mimic experimental fast folding of origami sheets, followed by clamping of a crease at a specific folded dihedral angle. Here we also adjust the clamped angle such that the overall magnitude of

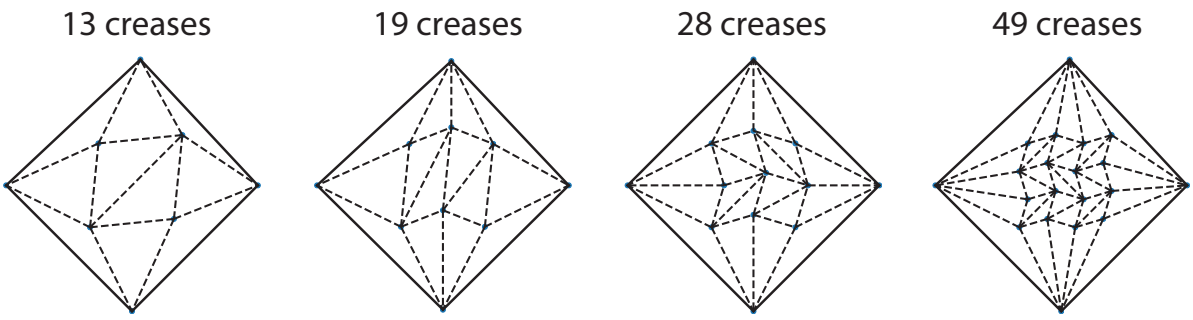

Figure 5.8: Origami Sheets used for training
The size of each sheet is determined by the number creases.

folding ρ remains fixed and different (discrete) folded structures may be compared more easily. Such fast folding was tested extensively [23], and found to obtain the same results as numerically solving the ODE of Eq. 5.8.

Origami sheets and applied force patterns

In this project we use specific self-folding origami sheets. These are triangulated thin sheets, chosen to have the property of self-foldability. As discussed above, a single vertex induces 3 mechanical constrains on the angles of creases surrounding it. Thus each vertex has to connect at least 4 creases or it would be locally rigid. On top of that, for a sheet to self-fold, it needs to have one global degree of freedom, so that the number of creases needs to be one more than the number of constraints.

A simple way of generating patterns meeting these requirements is shown in Fig. 5.8. These are 4 specific geometries used throughout this work as the sheets to be trained. Note that we label them according to their size, given by the number creases in each sheet. The number of creases in these sheets are $13, 19, 28, 49$ and the numbers of internal vertices are $4, 6, 9, 16$. Subtracting 3 times the number of vertices from the number of creases leaves us with one global degree of freedom for each of these sheets, as required.

The number of supported folded structures for these sheets grows exponentially with the number of internal vertices, such that these sheets can fold in approximately $2^4, 2^6, 2^9, 2^{16}$ distinct

ways [22, 81]. In fact, any sheet with these topologies (yet different geometries) will have a similar number of distinct folded structures. The exact details of the supported folded structures is dependent on the specific geometry, but we only require the existence of many distinct folded structures for the purpose of training.

These specific sheets, used for training classifiers throughout this work, are definitely not special. We attempted training classifiers using sheets with different geometries and obtained comparable results. In analogy to learning algorithms, the details of the sheet and its supported folded structures correspond to the family of models that the training protocol selects from. For origami, we believe the available classification models are given by merger of attractors of folded structures, supported by the sheet. Since the number of available models to choose from is exponentially large, we reason that the geometry of the sheet should play little role in the success of classification. Therefore, any self-folding origami sheet could be used for training classifiers.

The choice of force patterns applied to the sheets is constrained by the problem definition as training and tests sets. Still, there is usually freedom in how these forces are applied. For example, suppose we wish to train the 13 crease sheet of Fig. 5.8 on $2d$ force distributions, such as the spherical caps shown in Fig. 5.3. The training and test sets could thus be supplied as pairs of numbers, together with a label (blue\orange). A simple choice for training on such a data set is to pick two creases in the sheet and apply torques directly to these creases, as in Eq. (5.8). Here we utilize a different approach.

For an untrained sheet with homogeneous stiffness, it is known that all folded structures reside in the linear null space of the vertex constraint matrix C at the flat state [22]. Thus, forces applied in a direction within this null space are more 'natural' for the sheet, and in general cost much less energy due to face bending. We compute the span of the null space for each one of these sheets, and find that the dimension of the null space is $d_{NS} = \#_{\text{creases}} - 2\#_{\text{vertices}}$. Therefore the 13 crease sheet has a $5d$ null space, while the 49 crease sheets has $17d$ null space. Then, the training and test examples are mapped to forces in the null space as follows. For a $n - d$ data set, we choose n random orthonormal vectors in the null space. Each training\test example is mapped to a force

pattern by assigning every component to one of the random orthonoraml vectors. Now these forces can be directly applied to the sheet to facilitate the training protocol.

Training results in heterogeneous crease stiffness values that change the geometry of the folded structures, so that they do not strictly reside in the null space of the untrained sheet. Still, for the moderate heterogeneity developed during training, the observed folded structures are very close to the null space, such that the described mapping is still useful and practical.

Supplementary Note 2 - Training origami sheets

Learning rule

As discussed in the main text, self-folding origami sheets naturally give rise to complex mapping of force patterns to folded structures, with exponentially many structures supported by the sheet. The learning rule developed in this work is meant to modify that map by changing crease stiffness coefficients, such that only a small number of folded structures are retained, corresponding to the desired classes. Here we will define precisely how the learning rule is chosen and applied to the sheet in order to develop the desired mapping.

According to the specification of the classification problem, the trainer has no a-priori knowledge of the true underlying force distributions. Instead they are supplied with a list of labeled force patterns ('cats' and 'dogs'). These training examples are used to find a reference folded structure in the following way. We fold an untrained sheet with every 'dog' example in the training set and record the folding angles of the obtained folded structures. Then, a reference 'dog' structure $\hat{\boldsymbol{\rho}}_{\text{dog}}$ is defined as the average of all of these folded structures (normalized appropriately)

$$\hat{\boldsymbol{\rho}}_{\text{dog}} \equiv \frac{\sum_{\boldsymbol{F}\in\mathcal{F}^{\text{dog}}} \boldsymbol{\rho}_U(\boldsymbol{F})}{\| \sum_{\boldsymbol{F}\in\mathcal{F}^{\text{dog}}} \boldsymbol{\rho}_U(\boldsymbol{F})\|}, \tag{5.10}$$

with $\mathcal{F}^{\text{dog}}$ the set of 'dog' training force patterns and $\boldsymbol{\rho}_U(\boldsymbol{F})$ the folded response of the un-

trained sheet to force pattern $\boldsymbol{F}$. A similar reference state $\hat{\boldsymbol{\rho}}_{\text{cat}}$ is obtained for the 'cat' training examples. Crucially, once the reference structures are set for the untrained sheet, they are kept fixed throughout the training process. These reference structures are used to define the learning rule discussed in the main text. Suppose that during the training protocol, we choose a random 'dog' example $\boldsymbol{F}^{\text{dog}}$ and apply it to the sheet. The normalized resulting folded structure is written as $\boldsymbol{\rho}(\boldsymbol{F}^{\text{dog}})$. The learning rule then compares this folded structure to the reference structures defined above and the stiffness coefficients are modified as follows:

$$\begin{aligned} &\text{if } \boldsymbol{\rho}(\boldsymbol{F}^{\text{dog}}) \cdot \hat{\boldsymbol{\rho}}_{\text{dog}} > \boldsymbol{\rho}(\boldsymbol{F}^{\text{dog}}) \cdot \hat{\boldsymbol{\rho}}_{\text{cat}} : \quad \frac{dk_i}{dt} = -\alpha \rho_i^2(\boldsymbol{F}^{\text{dog}}) \\ &\text{else} \ : \qquad\qquad\qquad\qquad\quad \frac{dk_i}{dt} = +\alpha \rho_i^2(\boldsymbol{F}^{\text{dog}}). \\ &k_i \geq 0, \ i \in \text{creases} \end{aligned} \tag{5.11}$$

In essence, the learning rule checks whether the observed folded structure is closer to the 'dog' reference then to the 'cat' reference. If it does, the stiffness of creases that fold considerably in that structure is reduced, effectively reinforcing this force-fold mapping. An opposite modification occurs if the folded structure is far away from the 'dog' reference. A similar training rule is used when 'cat' forces patterns are applied, with the understanding that we wish to compare the resulting folded structure $\boldsymbol{\rho}(\boldsymbol{F}^{\text{cat}})$ to the 'cat' reference $\hat{\boldsymbol{\rho}}_{\text{cat}}$.

Assigning labels to folded structures

To begin with, we are given labeled force patterns, and an untrained sheet with many available folded structures. It is important to note that these folded structures are equivalent and not intrinsically labeled. Thus, as part of the learning protocol we must specify how to label these folded structures, and in particular which of them to call 'dog' and 'cat' (or 'blue' and 'orange'). A simple solution would be to choose 2 of the folded structures in advance and assign the classification labels to them. Unfortunately, this turns out to be too restrictive for a couple of reasons. First, the choice

may be far from ideal in the sense that these labeled folded structures are very different than the actual folded response of the sheet to the labeled force patterns. Furthermore, as the training process modifies the stiffness of different creases, the folded structures supported by the sheet change as well, either by moving around or disappearing altogether in saddle-node bifurcations [23]. We thus take a different approach to labeling folded structures, as detailed below.

Suppose we have trained a sheet for some time, and it now has a particular stiffness profile on its creases k_i. To find a folded structure of this sheet to be labeled 'dog', we apply each of the 'dog' training examples once, and record the discrete resulting folded structures due to all of them $\{\boldsymbol{\rho}(\boldsymbol{F} \in \mathcal{F}^{\text{dog}})\}$. We then count the training force patterns that folded into each one of the structures in this set. The folded structure that resulted from the largest number of training force patterns is chosen to be labeled as 'dog'. In case of a tie, e.g. two or more folded structures folding as a result of the same number of force patterns, one of these structures is randomly chosen to serve as the label. Thus, the labels for 'dog' and 'cat' are decided through simple plurality rules every time we compute the classification accuracy of the sheet. Note that force patterns may also fold the sheet into structures not labeled as either 'cat' or 'dog', in which case they count as failed classification. If both 'dog' and 'cat' labels are chosen to be associated with the same folded structure, a plurality rule between the two classes decides which class is labeled with that structure (i.e. whether more 'cat' or 'dog' force patterns folded into that structure), while the other is assigned with the runner up structure of that labels' plurality vote. Finally, if the sheet is overtrained to the point where only one folded structure remains, that structure is labeled as both 'cat' and 'dog', such that classification fails completely, by definition.

Effective cost function

In this work we have defined our learning rule as a supervised physical process modifying the stiffness coefficients of an origami sheet. It is interesting to compare this kind of learning protocol to more established learning algorithms originating in computer science and statistics. One important difference is that traditional learning algorithms are usually defined as an optimization problem,

where the function to be optimized (often called cost or loss function) incorporates the training data.

A simple example is linear regression, where the cost function is usually chosen as a least squares form, where the differences are taken between a linear model $h(x)$ and the observations y:

$$\begin{aligned} \text{Cost} &\equiv \sum_{d \in \text{data}} (h(x_d) - y_d)^2 \\ h(x) &= a_0 + a_1 x \end{aligned} . \tag{5.12}$$

The regression (or learning algorithm) then optimizes the cost function with respect to the model parameters $\boldsymbol{a} \equiv (a_0, a_1)$

$$\underset{\boldsymbol{a}}{\text{minimize}} \quad \text{Cost}(\{x\}, \{y\}; \boldsymbol{a}).$$

This optimization can be achieved in any number of ways, but a practically favored method (at least for more advanced algorithms like deep learning) is mini-batch stochastic gradient descent (SGD) [117]. In an extreme case, when the mini-batches are chosen to be of size 1, a single training example (x, y) is chosen at random in each step, and one computes the gradient (with respect to parameters $\boldsymbol{a}$) of the cost function defined with this example alone $\boldsymbol{G} \equiv \nabla_{\boldsymbol{a}}(h(x) - y)^2$. Now, training proceeds by modifying the parameters in proportion to the the gradient of this single example cost function

$$\boldsymbol{a} \to \boldsymbol{a} - \alpha \boldsymbol{G}, \tag{5.13}$$

where α is a scalar known as the learning rate. We may compare this single example SGD with our origami training protocol. It is relatively easy to see that our training rule (Eq. 5.11), once a standard wait time is chosen at the folded state, has the form of SGD, making it similar in essence

to other learning algorithms. To find out what effective cost function gives rise to the origami learning rule, we integrate Eq. 5.11 with respect to the stiffness coefficients

$$\begin{aligned} &\text{cost}_{\text{map}}(\boldsymbol{\rho}(\boldsymbol{F}^{\text{dog}})) = f \sum_{i \in \text{creases}} k_i \rho_i^2(\boldsymbol{F}^{\text{dog}}) \\ &\text{if } \boldsymbol{\rho}(\boldsymbol{F}^{\text{dog}}) \cdot \hat{\boldsymbol{\rho}}_{\text{dog}} > \boldsymbol{\rho}(\boldsymbol{F}^{\text{dog}}) \cdot \hat{\boldsymbol{\rho}}_{\text{cat}} : \qquad f = +1 \\ &\text{else} : \qquad f = -1 \end{aligned} \quad . \tag{5.14}$$

Similarly to the linear regression example, our origami training protocol attempts to minimize this derived cost function, one training example at a time. Inspecting this function, note that it is very similar to the energy of the torsional springs in the folded structure $E_{\text{Crease}}(\boldsymbol{\rho}) \sim \sum_i k_i \rho_i^2$. The difference is in the 'supervising factor' f that can be ± 1 whether the folded structure is accepted or not. We conclude that our origami training protocol is attempting to minimize the energy of accepted folded structures, while maximizing the energy of rejected structures.

Supplementary Note 3 - Using origami sheets to define classification problems

The force distributions classified in the main text are relatively simple. Both the spherical cap and the Iris data distributions can be well separated by a hyper-plane, a very simple decision boundary. It is interesting to study the type of decision boundaries naturally trainable in origami sheets – and whether they can be used to classify intrinsically high dimensional data.

There are many ways to obtain high dimensional distributions. Here we choose to study distributions derived from the folding maps of origami sheets. Consider a relatively simple sheet with 2 internal vertices (Fig. 5.9a). It is known that such sheets support 4 discrete folded structures, and that the linearized null space in which they reside is 3-dimensional. Therefore, if we sample random force patterns within this null space, we expect to see the sheet folding into 4 distinct struc-

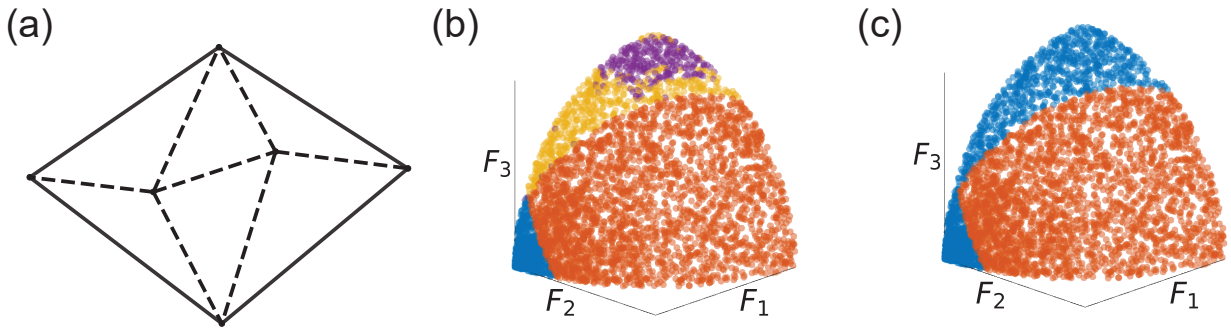

Figure 5.9: Defining force distributions using the force-fold mapping of an origami sheet
a) Origami sheets with 2 internal vertices support 4 discrete folded structures. b) Sample force patterns on a 2-sphere show the force-fold mapping (4 color coded regions). c) When some attractor regions are merged (here, blue, yellow and purple are merged), we obtain an intrinsically 2-dimensional separator surface between two classes of force patterns.

tures (color coded regions in Fig. 5.9b). The forces F_1, F_2, F_3 are assigned by randomly choosing Euler angles on the 2-sphere, and 3000 data points are sampled on the positive octant. Note that we sample normalized forces on the surface of a $2-$ *sphere*, such that the distribution of force patterns is actually 2-dimensional.

Now, suppose we wish to classify forces to 2 classes ('blue'\'orange'). A simple way to create 2 neighboring sets of points is to take the data of Fig. 5.9b and merge some attractor regions to create larger groups of points. In Fig. 5.9c, we merge the 'blue', 'yellow', and 'purple' folded structures to create one region we define as 'blue'. This process yields two distributions that are intrinsically 2-dimensional, and not naturally separable by a hyper-plane. Larger sheets can be similarly used to create force distributions in higher dimensional space.

With this process, we have access to a new variety of 2-way classification problems, on which we can try to train origami sheets using the training protocol described in the main text. Crucially, the sheet used to classify such distributions is different than the sheet used to derive the distribution. In other words, we ask if our training protocol can induce an origami sheet to mimic the force-fold mapping of another sheet.

Suppose we want to classify the distribution seen in Fig. 5.10a, derived form a 2-vertex sheet as described above. We wish to train a 13 crease sheet to classify this force pattern data. The

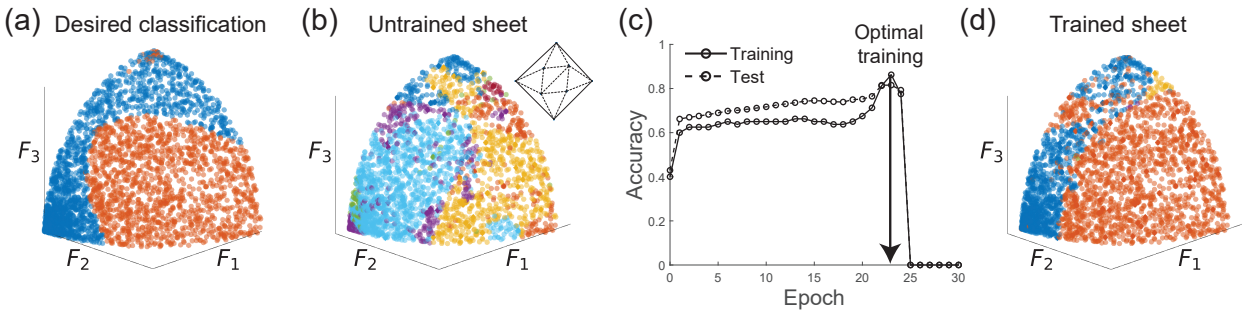

Figure 5.10: Training a sheet on a force distribution derived from a different sheet
a) Target classification, a sample distribution derived from a small, 2-vertex origami sheet. b) The force-fold map of an untrained 13 crease sheet is very different from the desired mapping. c) With training, the accuracy of classification improves and peaks at 82%. d) The optimally trained sheet has a complex decision boundary that resembles (but different than) the desired boundary.

untrained sheet has 2^4 discrete folded structures that do not align with the target distribution in any representation that we tested (Fig. 5.10b). The problem of classification here is to train this sheet to have just 2 folded structures with the right force-fold mapping as in the target distribution.

The target distribution is mapped to applied force patterns on the 13 crease sheet by the construction describe in Supplementary Note 1: choosing random orthonormal vectors in the null space of the 13 crease sheet and mapping the distribution as components of these vectors. We then randomly sample 20 'blue' and 20 'orange' force patterns, marked as diamonds in Fig. 5.10, to serve as the training set. As we train the sheet, the classification accuracy improves dramatically and reaches a maximum of 82% (test accuracy) after 23 epochs (Fig. 5.10c). To qualify the classification better, we look at the classification results corresponding to the maximal accuracy at epoch 23 (Fig. 5.10d). We observe that the trained decision boundary resembles the desired boundary, so that the training protocol indeed produced a reasonable classification.

Note a few artifacts that still remain in the trained map: 1) there are 3 folded structures left, rather than 2 (a small third color coded region exists, labeled yellow), 2) a second orange region appeared inside the bulk blue region, emphasizing that the decision boundaries between folded structures in sheets are generally *not* hyper-planes. We conclude that origami sheets can be trained to classify distributions derived from other sheets, that are intrinsically higher dimensional than

the problems discussed in the main text. We leave questions of the sheet size and the complexity of decision boundaries to future studies.

Supplementary Note 4 - Transforming Iris data to applied forces on sheets

The Iris data set [125] classified in the main text is a classical problem for classification. In this work we are able train an origami sheet to correctly classify two species of Iris (I. Versicolor, I. Virginica) at an accuracy of 91%. Here we discuss how the Iris data is used to generate training and test sets of applied force patterns to be used on origami sheets.

Each Iris example in the data set is given as a vector with 4 features (components): sepal length, sepal width, petal length, petal width. These length measurements are all given in *cm*. In addition to these measurements, each Iris is labeled as one of the Iris species in the study. To generate force pattern sets from this data, we would like the different measurements for each Iris example to be components of force vectors in the null space of the origami sheet, as described in Supplementary Note 1. However, the raw Iris data is not suited for this purpose due to two reasons. The dimensionful measurements of Iris lengths, if directly translated to forces, would be far too great for our sheets and will cause it to fold too much and cause the sheet faces to collide. More crucially, sepal and petal lengths tend to be considerably larger than their widths, and the same goes for the variance of these variables. This will causes the width variables to be perceived as less important in the training protocol, and have a negative effect on the classification results.

Fortunately, diverse data like this is an issue regularly faced by learning algorithms, and it is generically solved by applying an invertible transformation to the data. The transformed data is then better suited for the learning algorithm in use. A typical example of such a transformation in data sets is to normalize each feature (divide by the mean of that feature) and translate it such that the mean of the transformed data is 0. This transformation is especially useful for classification algorithms like logistic regression, where the different features have different dimensional units.

In our case however, the standard transformation above is not useful, due to a particular property of origami sheets, namely their Z_2 symmetry. If forces $\boldsymbol{F}$ are applied to the sheet and it folds into a state $\boldsymbol{\rho}$, then folding the same sheet with forces $-\boldsymbol{F}$ will result in a state $-\boldsymbol{\rho}$. This is true for any self-folding origami sheet, regardless of the stiffness profile on its creases. This property cannot be changed by training the sheet. Thus, force patterns of opposite sign and different labels cannot be correctly classified. A simple way to avoid this issue is to limit the force patterns to reside in a restricted part of force space. We choose to limit the distributions such that the transformed Iris data will all be in the positive 4-hyperoctant.

In addition, we want the data to span as much as possible of the positive hyperoctant. This will increase the expressive of our training protocol, as more discrete folded structures would become available if the applied force patterns are more diverse. We thus need to transform the Iris data to be all positive, and stretch it such that all features have similar variance.

To achieve these goals we apply the following linear (invertible) transformation to the Iris data of the Versicolor and Virginica species. Suppose an Iris example is given as a vector $\boldsymbol{x}$ (where the components are sepal length, sepal width, petal length, petal width in this order). The vector is transformed by

$$\boldsymbol{x}^* = A\boldsymbol{x} + b$$

$$A = \begin{pmatrix} 0.264 & 0 & 0 & 0 \\ 0 & 0.580 & 0 & 0 \\ 0 & 0 & 0.303 & 0 \\ 0 & 0 & 0 & 0.836 \end{pmatrix}, \quad b = -0.880. \tag{5.15}$$

Then the transformed vector is used to define the force patterns applied to the origami sheet, as described in Supplementary Note 1. After training is concluded, the transformation can be inverted to relate the origami classification results with the original Iris data, as shown in the main text.

Chapter 6

Learned multi-stability in mechanical networks

Materials design is generally predicated on knowing the desired material behavior at the time of design. If an adaptable material with multiple behaviors is desired, all potential desired behaviors are usually specified in advance. As a result, we can optimize design parameters compatible with all of the specified desired behaviors. Among mechanical metamaterials, such design has been fruitfully used to create materials that switch from being soft to stiff, transparent to opaque or energy absorbing to elastic, by simply switching between different stable geometric states of the material [12, 13, 17, 138, 139, 140, 141, 142, 143, 144, 145].

Here, we explore an alternative approach, where a material *learns* desired behaviors on the fly by physically experiencing such behaviors in sequence, e.g., by being held in each desired state for a period of time. Such a learning framework for materials offers many complementary strengths to the conventional design framework. For example, the precise behaviors needed can be inferred from the actual conditions of use in real time, instead of an *a priori* specification. New functionalities can be gained during, and due to, use. Such benefits have made learning a powerful framework in neuroscience and artificial neural networks, but this framework is relatively

unexplored in the context of materials [7, 15, 146].

However, learning in the context of materials presents challenges in addition to such opportunities. In the learning framework, the desired behaviors are not all known ahead of time but presented sequentially. Thus material parameters to encode each desired behavior must be chosen independently without knowledge of future desired behaviors. Most critically, each stored behavior or state needs to survive the parameter changes due to the subsequent learned behaviors and not be overwritten by them. It is not clear what kinds of material properties and interactions would allow such sequential learning of multiple behaviors.

In this work we contrast the requirements for design and learning of multiple stable states in a simple elastic network. In the design model, we search over all spring constants on a computer to stabilize a set of states that are specified beforehand. In the learning model, the desired states are learned in sequence by example, placing the material in each of these states for a period of time. During this time, stabilizing elastic rods or springs with a rest length grow between particles within some distance in space, mimicking the seeded growth of microtubules [147] or self-assembling DNA nanotubes [148]. Thus, in contrast to design, the 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 [7, 146].

As a direct consequence, we find that successful learning requires non-linear elasticity of a specific type. Parameterizing the elastic energy of springs in the network as $E \sim x^{\xi}$ for large extensions x, we find that our design procedure is optimal for $\xi \approx 2$ (Hooke's law) but learning requires $0 < \xi \leq 1$. Such nonlinear springs have been demonstrated using metamaterial designs [149, 150]. We relate this distinction to the way springs are unequally strained in a learned state – springs learned for that state are nearly unstrained while all other springs are highly strained. Such 'sparse' strain profiles are stabilized by $\xi \leq 1$ springs but not for $\xi > 1$.

We establish these results by relating spring non-linearity to Bayesian priors used in statistical regression; such priors can pick out sparse solutions to equations in which some variables are exactly zero. Much in the way Bayesian priors dictate sparsity in statistical regression, the non-

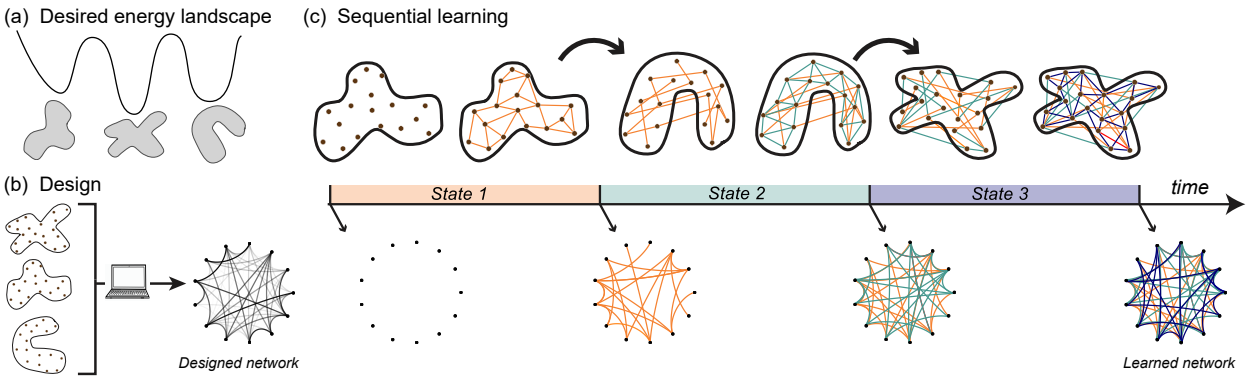

Figure 6.1: Designing vs learning multiple states
(a) We seek to create an elastic network with specific stable configurations. (b) In the design approach, all desired states are specified beforehand and then network parameters (connectivity, spring constants, rest lengths) are optimized to stabilize these states. (c) In learning, the material is physically placed in the desired states in sequence and the network grows incrementally according to the local geometry of that state (Eq. 6.1). Hence information about each desired state is localized to only a fraction of network links (different colors). For learning to succeed, network changes due to learning state 2 should not interfere with the stability of state 1 or vice-versa.

linearity of springs dictates that information about each learned state is localized in the material. 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.

6.1 Results

We seek to create an elastic network of springs connecting N particles in 2 dimensions, such that the network has M desired stable states (Fig. 6.1a). Each desired stable 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 our design model, 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. 6.1b); see Supplementary Note 1 for details.

In the learning model, desired stable states are acquired by sequentially placing the material in the desired configurations (Fig. 6.1c). When 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 separa-

tion 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. Such elastic elements that grow between specific sites are found both in living systems (e.g., microtubules growing between centrosomes and centromeres [147, 151]) and in synthetic systems (e.g., self-assembling DNA nanotubes [148] growing between seeds).

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,

$$\frac{dk_{ij}^{\text{eff}}}{dt} = k_0 f(r_{ij}). \tag{6.1}$$

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., rods are unstretched in the desired state. In simulations, we pick f to be a step function of range R, $f(r < R) = 1, f(r > R) = 0$.

Equation 6.1 describes the learning rule for this material; the effective spring constant and rest length between two particles i, j is determined by the geometric configurations experienced by the material and the amount of time spent in each configuration. When the material is deformed and held in a second distinct configuration $\mathbf{x}^{(2)}$, additional rods start growing between the particles according to their positions in the new configuration. In some cases, two particles can be joined by multiple springs with different rest lengths.

6.1.1 Linear and non-linear elasticity

We ran the design and learning algorithms using rods with linear Hookean elasticity, i.e., with elastic energy $E_{ij} \sim k_0 s_{ij}^2$ when strained by s_{ij}. The design algorithm, when run on a pair of randomly generated desired states $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ of 10 particles, resulted in an elastic network with two stable minima that resemble $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$, as seen Fig. 6.2a. These states can be retrieved by any initial condition within wide attractor regions around $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}$.

In contrast, learning the same two states $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}$ with linear springs fails (Fig. 6.2b); the two

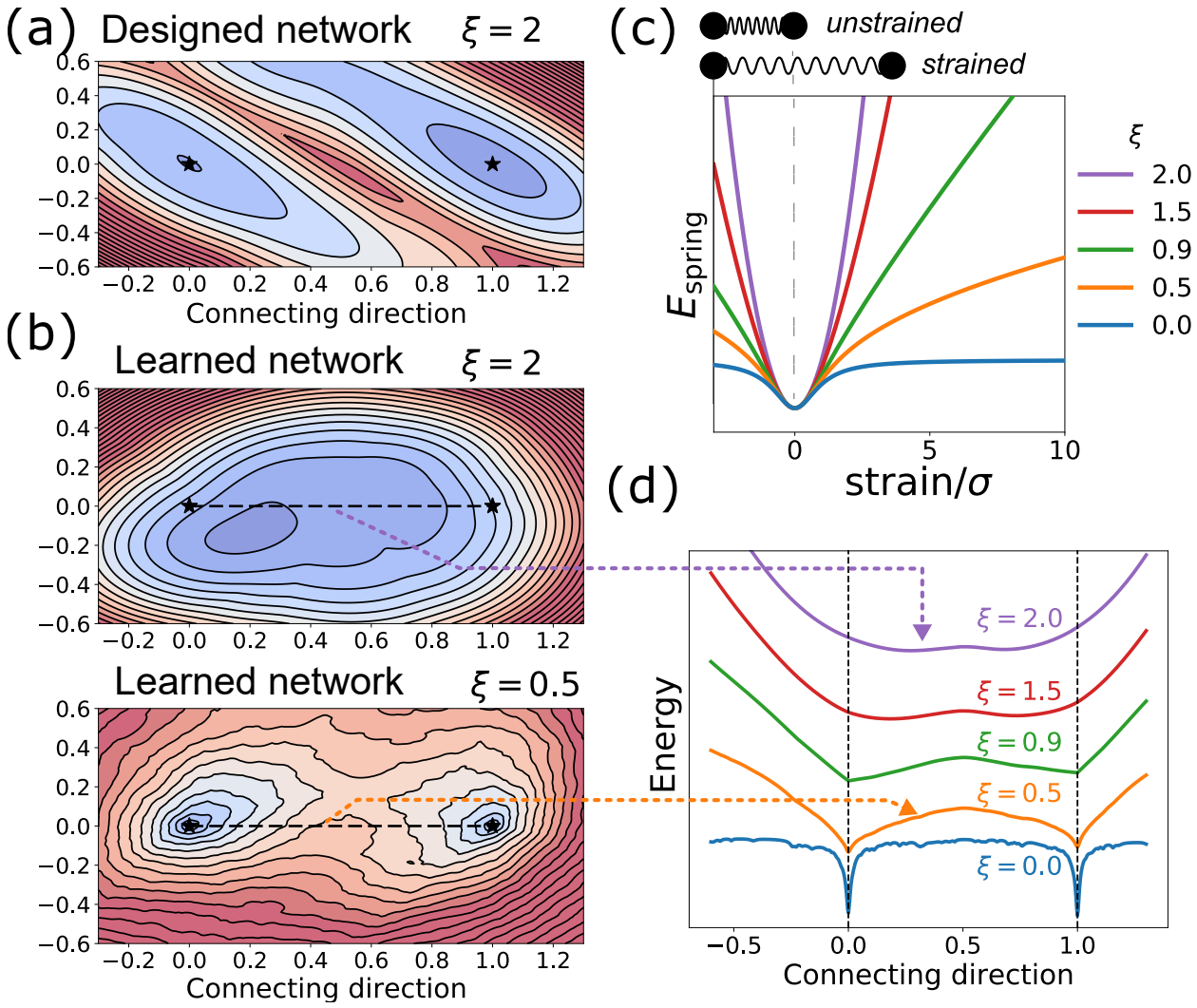

Figure 6.2: Non-linear interactions are essential for learning multiple states in sequence (a) Energy landscape of a designed network with linear ($\xi = 2$) springs successfully stabilizes desired states (black stars). (b) In the learned network, linear ($\xi = 2$) springs learned for each desired state destabilize the other state, but non-linear ($\xi = 0.5$) learned springs stabilize both desired states. (c,d) Repeating learning for non-linear 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 non-linear $\xi \leq 1$ springs.

desired states are not stable minima of the learned network. The rods grown to encode state $\mathbf{x}^{(1)}$ destabilize, or overwrite, state $\mathbf{x}^{(2)}$ and vice-versa. Initial conditions near either $\mathbf{x}^{(1)}$ or $\mathbf{x}^{(2)}$ relax to new minima very different from $\mathbf{x}^{(1)}$, $\mathbf{x}^{(2)}$.

Why do linear springs allow stabilization of multiple states with design but not with sequential learning? In design, the desired configurations are known ahead of time and so each spring's parameters can be chosen cognizant of all desired configurations. In fact, one can check that changing one of the desired states, e.g., $\mathbf{x}^{(1)} \to \mathbf{x}^{(1)} + \delta\mathbf{x}^{(1)}$ changes stiffness k_{ij} and rest length l_{ij} for *all* springs. In this sense, information about each desired state is stored in every spring.

However, in a learning model capable of acquiring arbitrary stable states in sequence, the parameter changes made to store a state $\mathbf{x}^{(m)}$ cannot depend on the details of future desired states [152], and indeed, in this model, does not depend on past encoded states either. That is, changing a desired configuration, e.g., $\mathbf{x}^{(m)} \to \mathbf{x}^{(m)} + \delta\mathbf{x}^{(m)}$ changes the spring parameters k_{ij}, l_{ij} only for springs grown while learning state m. Thus, information about each stored state is confined to a subset of springs.

Consequently, to stabilize a state $\mathbf{x}^{(m)}$, the elastic dynamics should only attempt to minimize strain to zero in a subset of all springs while leaving all other springs stretched arbitrarily as needed. However, the mechanics cannot possibly know which subset of springs was learned to stabilize a particular state $\mathbf{x}^{(m)}$ and thus which subset to satisfy.

A clue to solving this problem comes from sparse regression [153, 154]. As an example, consider an under-determined problem $A\mathbf{s} = b$ 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 non-zero, we can find such 'sparse' solutions $\mathbf{s}$ by adding a 'Bayesian prior' $\|\mathbf{s}\|^{\xi} = \sum_i s_i^{\xi}$ to the least squares loss function,

$$E = \|A\mathbf{s} - b\|^2 + \|\mathbf{s}\|^{\xi} \tag{6.2}$$

and then minimizing E [153, 155]. If $\xi \leq 1$, such a Bayesian prior $\|\mathbf{s}\|^{\xi}$ biases the search towards solutions $\mathbf{s}$ in which some elements of $\mathbf{s}$ are strictly zero while others are non-zero (i.e.,

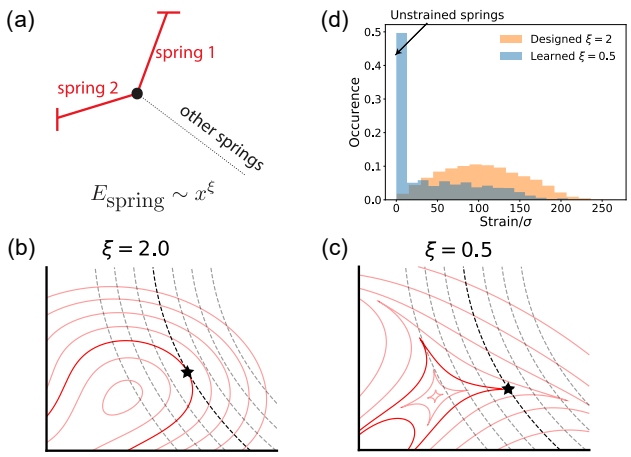

Figure 6.3: Non-linear springs apply a Bayesian prior to the strain distribution
(a) The energy of the two red springs is represented by red contours, that of all other springs by black contours. The whole system's energy minima will be at points where these contours are tangent to one another. (b) If $\xi > 1$, 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 one of the red springs is unstrained. (d) Typical stable states of a large $N = 100$ network have many unstrained springs if and only if $\xi \leq 1$.

'sparse' solutions). We emphasize that the Bayesian prior s^{ξ} contains no information about which components of s are to be set to zero; rather, it biases regression towards such solutions and away from generic solutions in which all entries of $\mathbf{s}$ are non-zero.

We employ a similar strategy here by identifying $\mathbf{s}$ above with the vector of strains in different springs. Let us assume that the network spring energies take a non-linear form,

$$E(s) \sim k_0 \frac{s^2}{(\sigma^2 + s^2)^{1-0.5\xi}}, \tag{6.3}$$

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 parameterizes the non-linearity (Fig. 6.2c); $\xi = 2$ 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 keep the model realistic, reflecting practical realizations of non-linear $\xi < 2$ springs [149, 150]; our results below hold for $\sigma \to 0$ as well. See Supplementary Note 2 for details.

We repeated the same learning procedure on the same states as earlier - but with non-linear springs $\xi < 2$. While the results for $1 < \xi < 2$ are qualitatively similar to linear springs $\xi = 2, \xi < 1$ shows qualitatively different results – learning succeeds in stabilizing multiple states (Fig. 6.2b,d).

How do we understand this result? It is clear that forces due to $\xi < 1$ springs diminish with strain and thus weaken the effect of strained springs that code for other states. However, the analogy with Bayesian priors goes further by explaining the sharp change in behavior at $\xi = 1$ due to the non-analytic nature of s^ξ. Following work in sparse regression [153], in Fig. 6.3b,c, we plot the energy contours for the red springs shown, where the two red springs have incompatible rest lengths. The constant energy contours are cusped for $\xi < 1$ but not $\xi > 1$. At the cusps, one of the two red springs is completely unstrained while the other contains all the strain. When minimized in conjunction with other springs (dashed black contours), minima are exceedingly likely to be at cusps for $\xi < 1$, where strain is localized to one spring.

Thus, non-linear $\xi < 1$ springs stabilize states with bimodal strain distributions - some springs are highly strained while others are unstrained. To complete the analogy with sparse regression, note that the energy of the system in Fig. 6.3 resembles Eq. 6.2. Let $\mathbf{F}^{ext}$ represent forces on the particle in Fig. 6.2a due to the black springs (assumed constant for simplicity). In the limit of small core sizes $\sigma \to 0$, the red spring energies are given by $E(r) = k(r - l)^\xi \equiv ks^\xi$, so that the total energy of the subsystem shown is,

$$E = -\mathbf{F}^{ext} \cdot \mathbf{x} + k \sum_{\text{red}} s_a^\xi = -\mathbf{F}^{ext} \cdot \mathbf{x} + k||\mathbf{s}||^\xi, \tag{6.4}$$

where $||\mathbf{s}||^\xi$ is the ξ-norm of the strain vector $\mathbf{s}$ for the red springs. The non-linear elastic energy has the analytic form of sparse regression, Eq. 6.2, and thus one of the red springs is unstrained in each stable minimum. Note that the springs now play a dual role, both providing the equation that is to be solved (the equivalent of $A\mathbf{s} = b$ in sparse regression), and providing the bias towards a bimodal strain distribution.

To test this analogy in larger elastic networks, we let a $N = 100$ particle network learn two

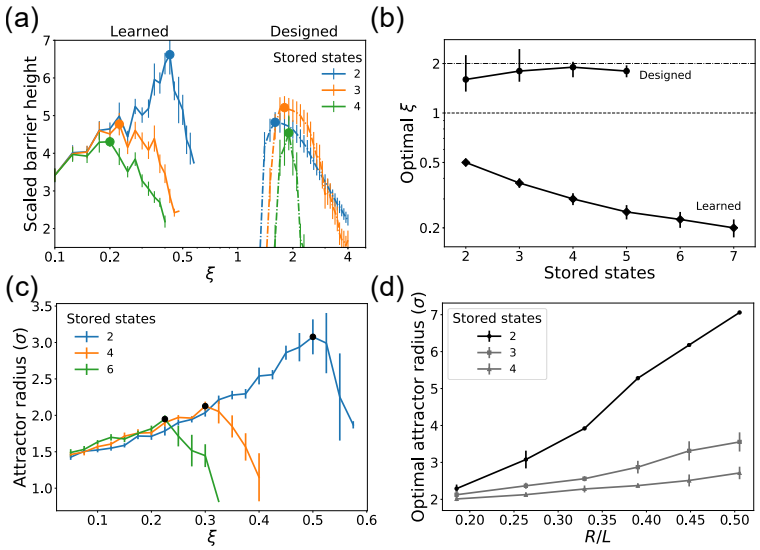

Figure 6.4: Optimal non-linearity for learned and designed states
(a-b) Barrier heights around designed states are highest for $\xi^* \approx 2$ (linear springs) but highest for learned states at a specific non-linearity $0 < \xi^* < 1$. Further, learning more states requires stronger non-linearity ξ^*. (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. 6.1, lead to larger attractor basins (see SI for details). L is the system length.

distinct states, and measured the strain in each spring after relaxing to one of the states (Fig. 6.3d). For non-linear springs $\xi < 1$, 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 designed minima with linear springs $\xi = 2$, for which all springs are strained.

6.1.2 Optimal non-linearity

The quality of both learning and design 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 [156, 157, 158].

We find that quality of designed and learned states, as measured by barrier heights, is highest at distinct ξ^*; see Fig. 6.4a,b. The quality of designed states, for our simple design algorithm, is optimal for linear springs $\xi^* \approx 2$ and is relatively insensitive to the number of designed 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 barrier heights (Fig. 6.4c). See Supplementary Note 3.

Much as in sparse regression [159, 160], the optimal ξ^* for learning can be understood as a balance of two factors – sparsity (smaller ξ) and convexity (larger ξ). Smaller ξ leads to more sharply cusped energy contours in Fig. 6.3c and thus a stronger bias towards bimodal strain distributions with zero strain in some springs (i.e,. sparsity). However, smaller $\xi \to 0$ also leads to vanishing restoring forces outside the immediate vicinity of the unstrained configuration, creating a 'golf course' landscape with vanishing attractors. Thus while smaller ξ locally stabilizes each desired minimum using bimodal strain distributions, larger ξ enlarges the attractor basin, making these minima easier to find. Similar considerations in canonical sparse regression problems select $\xi^* = 1$ as an optimal choice [153].

The radius of spring connection R plays an important role in setting the optimal ξ^* value. We observe that the additional stabilizing contributions of the springs afforded at larger R facilitates the optimal stabilization of the system at higher ξ^*, and thus with attractors of larger size, as seen in Fig. 6.4d (for more information see Supplementary Note 4). L is the length scale of the system.

6.1.3 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 [161], and try to teach an elastic network to recognize the digits '0' and '1' from examples of these digits.

We trained the 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 1-d configuration of 400 particles by interpreting each pixel's gray-scale value as a particles position in the interval $[0, 1]$. The particles in such a state are connected by elastic rods according to the learning rule in Eq. 6.1. For $\xi < 1$, we find that the training generally creates two distinct large attractors corresponding to an idealized 0 and 1 respectively (Fig. 6.5d).

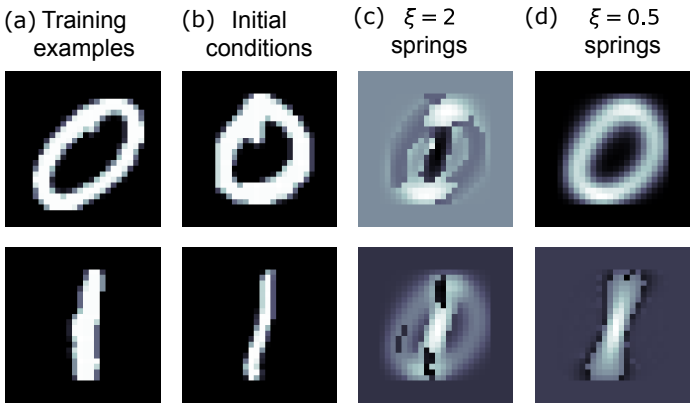

Figure 6.5: Elastic networks learn to recognize handwritten digits
(a) Images representing two particle configurations that we wish to stabilize (adapted from MNIST). The 400 pixel gray-scale values in each image are interpreted as positions of 400 particles in 1-dimension. We learned a non-linear spring network using 5000 randomly drawn examples of 0s and 1s 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 an unique uninterpretable state. (d) In contrast, non-linear 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.

We then test the network by using novel unseen examples of 0s and 1s from MNIST as initial conditions 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 non-linear $\xi \leq 1$ elastic network learns states 0 and 1 with sufficiently large attractors to accommodate the typical handwriting variations seen in the MNIST database.

6.2 Discussion

In this work we contrasted a design and a learning framework for creating multi-stable elastic networks. We found that continually learning novel states without overwriting existing states requires a specific non-linear elasticity $\xi \leq 1$. The learning model here relies on spontaneous growth of stabilizing rods between nearby nodes, a behavior displayed by microtubules [147], DNA nanotubes [148] and other such seeded self-assembling tubes [162, 163, 164].

The non-linearity ξ plays a unique role as a material design parameter. Most material parameters (e.g., l_{ij}, k_{ij} of springs here) encode information about desired states. But ξ encodes an assumption about how information about desired states is distributed among parameters l_{ij}, k_{ij} of different springs. Learning localizes 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 non-linearity 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. 6.2); the network has 2-d spatial geometry absent in Eq. 6.2 and is more closely related to (unsolved) sparse reconstruction of 2-d maps from pairwise distances between cities [165]. Consequently, we can explore how physical parameters with no analog in Eq. 6.2, such as the maximum range of learned interactions R (Fig. 6.4d) and spatial correlations between stored states, affect the optimal non-linearity ξ (Supplementary Note 4).

Learning and design have complementary strengths, as seen before in neural networks and spin glasses. For example, Hopfield [166] introduced neural networks that can learn arbitrary novel memories in sequence using a biologically plausible 'Hebbian' learning rule. Gardner [167] showed that the same model has a higher memory capacity if we assume an optimally designed network in lieu of learning. However, Gardner's network can be designed only when all desired memories are known — and must be redesigned from scratch to include new memories.

Similarly, in materials, design might be sufficient if all desired states are known beforehand and unlimited computational power is available, since design allows optimization over all design parameters. In contrast, learning is a physically constrained exploration of the same design 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. We hope the simple mechanical model studied here will stimulate further work on realistic learning rules that allow materials to acquire new functionalities on the fly.

6.3 Supplementary Notes

Supplementary Note 1 - Design of multiple stable states with linear and non-linear 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

$$E(\{\mathbf{x}\}) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=i+1}^{N} k_{ij}(r_{ij} - l_{ij})^2, \tag{6.5}$$

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. 6.5 with respect to node positions to zero:

$$0 = \partial_{\mathbf{x}_a} E = \sum_{i=1}^{N}\sum_{j=i+1}^{N} k_{ij}(r_{ij} - l_{ij})\frac{\partial r_{ij}}{\partial \mathbf{x}_a}. \tag{6.6}$$

This procedure gives Nd equations that have to be satisfied simultaneously for the Nd node coordinates. Note that Eq. 6.6 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 $d = 1 \rightarrow r_{ij} = |x_i - x_j|$). Due to the nonlinear relation of r_{ij} to x_i, x_j, multiple solutions $\{\mathbf{x}^\star\}$ can

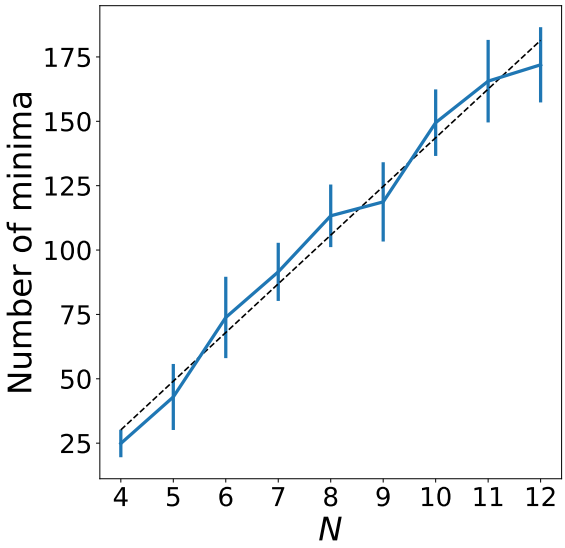

Figure 6.6: Number of stable configurations in a network of linear springs grows linearly with the size of the system

satisfy Eq. 6.6 simultaneously. 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. 6.6).

These multiple minima in the potential energy landscape, if moved around, could be utilized to program 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 tough Eq. 6.6 is nonlinear in node positions $\{\mathbf{x}\}$, it is linear in both k_{ij} and $a_{ij} \equiv k_{ij} \cdot l_{ij}$. Suppose we want to solve the system of equations 6.6 for M different node configurations denoted by $\{\mathbf{x}\}^m$, giving rise to distance matrices r_{ij}^m. Solution to such linear systems of equations can generally be found if the number of equations (NdM) is less than the number of variables ($0.5N^2d$). To design linear springs with multiple desired stable points, we thus numerically solve Eq. 6.6 simultaneously for 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.

The particular algorithm discussed above is only defined for linear springs with $\xi = 2$, as defined in the main text. Still, a design protocol for spring-node systems with any value of non-linearity ξ is possible. With non-linear springs the force balance of Eq. 6.6 becomes:

$$0 = \partial_{\mathbf{x}_a} E \sim \sum_{i=1}^{N} \sum_{j=i+1}^{N} k_{ij}(r_{ij} - l_{ij})^{\xi-1} \frac{\partial r_{ij}}{\partial \mathbf{x}_a}, \tag{6.7}$$

which is unfortunately non-linear 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. 6.7 over the design 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 designed 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 $0.5N^2d$ 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-hard nature of the design problem.

Supplementary Note 2 - Energy model for nonlinear springs

The main text establishes that to enable the learning paradigm to store multiple stable states in an elastic networks, 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 $F \sim (r - l)^{\xi-1}$. 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 $r = l$. 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 $|r - l| < \sigma$, and non-linearly otherwise. If we define a non-dimensional strain $u \equiv (r - l)\sigma^{-1}$, the energy of such a spring can can be written as:

$$E(u) = \frac{1}{2} k\sigma^{\xi} \cdot \frac{u^2}{(1 + u^2)^{1-0.5\xi}}, \tag{6.8}$$

with r the spring length, k stiffness, 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 non-linearity 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 regularization. The extreme limit $\xi = 0$ defines springs whose energy is a Lorentzian. Outside the core region, such springs exert forces that diminish quickly as $F \sim u^{-1}$. In general, the force due to the nonlinear springs is

$$F(u) = k\sigma^{\xi-1} u \cdot \frac{1 + 0.5\xi u^2}{(1 + u^2)^{2-0.5\xi}}. \tag{6.9}$$

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. 6.7. 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

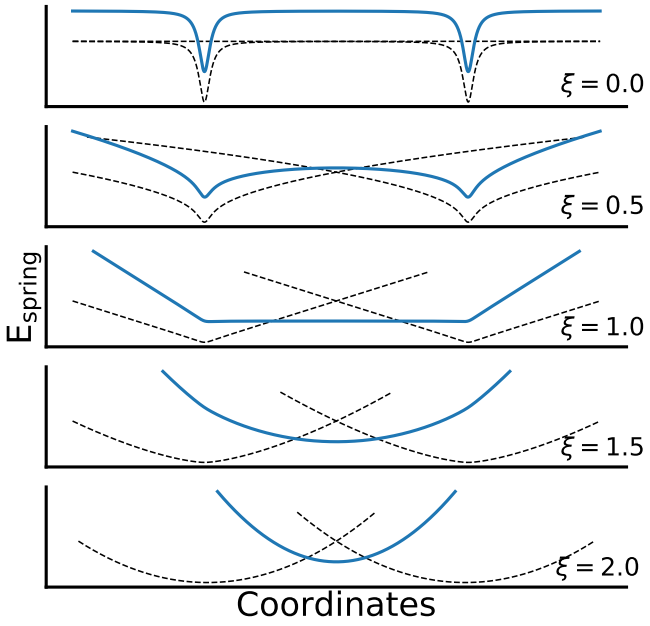

Figure 6.7: The sum energy of two springs goes through a transition at $\xi = 1$
The energy minimum of each spring is preserved for $\xi < 1$, while these minima are overwritten for $\xi > 1$. In essence, the information on minima of $\xi < 1$ springs is stored with each individual spring. (Black dotted 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).

range) enable the learning paradigm described in the main text. In 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 6.7 clarifies that springs whose force grow with strain are completely unfit for this purpose.

Supplementary Note 3 - 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 section describes some of the technical aspects involved in simulating these networks and deducing their properties. The codes to produce and study the elastic networks is implemented in Python and available upon request.

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 multi-stable 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 the paradigm studied (design, learning).

For the study of design, 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 6.6 in Supplementary Note 1. Construction of fully connected designed networks with non-linear springs ($\xi \neq 2$) is facilitated by optimizing forces at the desired stored states (Supplementary Note 1).

System 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, as outlined in the main text. All springs in this paradigm have the same spring stiffness k, core size σ and non-linearity 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.

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 $\frac{\|\vec{X}_*^{(m)} - \vec{X}^{(m)}\|}{2N} < \sigma$, 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 Supplementary Note 4 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 non-local 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 towards 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 (~ 0.5) of the typical force within the core distance σ. The inclusion of this force (gradient) requirement gives rise to an optimal non-linearity value $0 < \xi < 1$ for learned states, as shown in the main text.

Supplementary Note 4 - Stability of learned states

In the main text we established the usefulness of learning with non-linear springs as a means of programming multiple stable states into an elastic network. In this section 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.

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

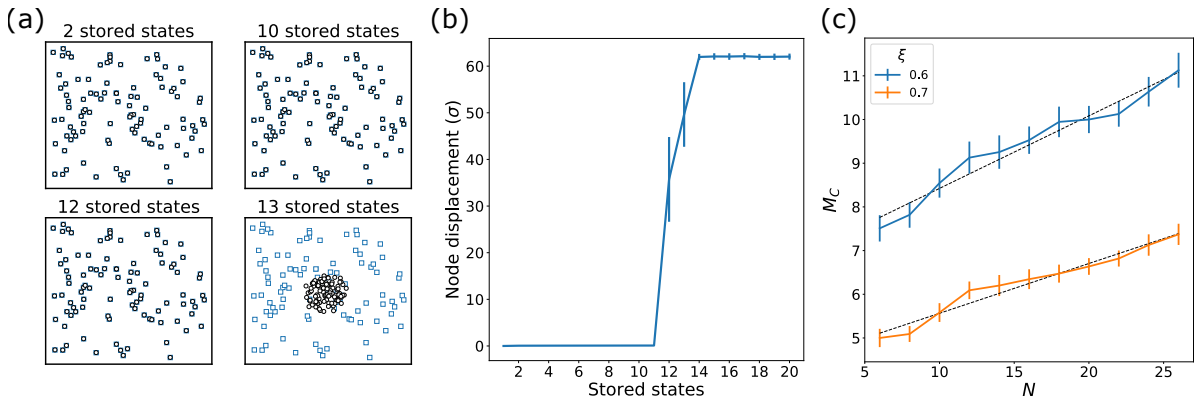

Figure 6.8: Programming stored states using the learning paradigm exhibits finite capacity a) Each stored state is affected by springs associated with other states. Initially the new springs have a small effect and the state remains a stable attractor. However, eventually states destabilize due to the 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 statess are simultaneously encoded ($N = 100$, $\xi = 0.6$). b) When node displacement is averaged over stored states, we observe a sharp failure of all stored states close to a specific load, defined as the *capacity* (12 – 13 states in this case). c) Capacity scales linearly with system size N.

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 6.8a 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. 6.8b). This capacity is well-defined and observed to depend on the parameters of the system, such as size N and non-linearity 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 $F_S \sim N$. 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 $F_{DS} \sim \sqrt{N \times M}$. The capacity of the system is reached when the magnitude of the destabilizing force is equal to that of the stabilizing force, so that

$$F_{DS}(M_C) \sim F_S \rightarrow M_C \sim N. \tag{6.10}$$

The capacity of a learning network is expected to scale linearly with system size, similarly to other Hopfield-inspired learning models [166]. This prediction was tested in networks with of sizes $N = 6 - 26$ and for several values of the non-linearity ξ. Results shown in Fig. 6.8c 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 non-linearity to be of lesser interest, as other metrics for quality of encoding (barrier height and attractor size), discussed in the main text, are more important for the robustness of learning.

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 Maxwell condition [168]. Then, a stable state of the over-constrained 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, usually results in stable states surrounded by higher energy barriers.

This argument suggests an intriguing possibility, that increasing connectivity in learned network may improve the stability and quality of the state storage. Such an outcome is possible as the act of adding more non-linear ($\xi < 1$) 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.

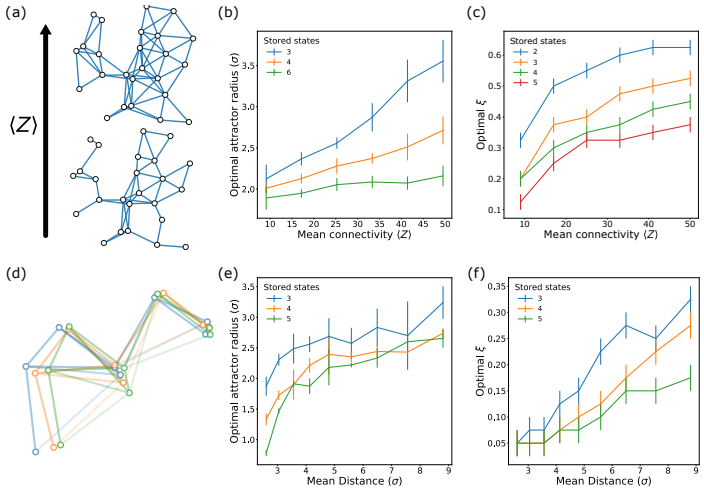

Figure 6.9: Effects of node connectivity and state similarity on the quality of encoded states (a-c) Connectivity between nodes $\langle Z \rangle$ increases with the effective connection radius R. We find that the more internally connected a state is, the larger its attractor size, and higher the optimal value of the non-linearity parameter ξ ($N = 100$). (d-f) Trying to store similar states is more difficult than random states. When the mean distance between nodes in successive states are small, attractors basin are also small, and successful encoding requires small values of ξ and flat potentials ($N = 100$).

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 our learning paradigm, connectivity is controlled by the effective radius of rod growth R defined in the main text. 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 Figure 6.9(a-c) verify that the quality of state storage, as measured by the attractor size of states, improves with their connectivity.

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 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. Figure 6.9(d-f) shows 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).

Chapter 7

Discussion

In this work we demonstrated and compared design and learning approaches for programming mechanical systems to perform defined tasks. Both approaches allowed us to achieve the desired results for self-folding origami and elastic networks. Studying these systems using the lens of simplified energy models was indeed a fruitful choice, as the energy landscape viewpoint often facilitated the definition of the desired goals (i.e. desired functionality was defined by desirable features of the energy landscape). The best example in this work is how multi-stability in elastic networks was envisioned as an energy landscape with minima at specific points in configuration space. Such a graphical representation suggests a design paradigm that minimizes forces at these points, and a growth (learning) framework that automatically chooses these points as minima.

We have discussed the benefits and drawbacks of design and learning in both mechanical systems, and arrive at some broad conclusions. Design can often be defined as a one-step process, optimizing a certain design function to achieve the desired goal. Thus, a design problem can often be solved on a computer without the need of constructing actual physical systems. Moreover, a design process follows a *rational* understanding of the system at hand and the task it needs to perform. In this work, our design approaches optimized rationally defined function, like the energy of folded states in origami, or the forces at stable elastic network states. In such cases, the solution to the optimization problem has a *clear physical interpretation*. For example, the sheets of chapter

4 fold successfully into the desired states because they have the lowest energy in the landscape, ensuring they survive the saddle-node bifurcations. These advantages have made design the de facto approach for engineering mechanical systems. However, design has drawbacks as well. In particular, designing large systems tends to be a computationally (NP-)hard problem, as it entails computational optimization of generically non-convex functions. Furthermore, as shown in chapter 6, design usually does not account for future unforeseen uses for the system. If the system needs to perform a new task on top of the previous ones, it generally needs to be completely redesigned.

Learning in physical systems, a concept not well studied so far, can give rise to impressive advantages compared to design. Learning systems can be modified according to use cases in response to external inputs. Such adaptive systems can be trained to perform tasks, e.g. classification of forces, even if the task is not clearly defined. For example, origami can correctly classify 'cats' and 'dogs' even if it did not train by observing all of these cats and dogs, and without any specification of what cat and dog actually are. Thus, just as in neuroscience and computer science, learning is a powerful approach that can program tasks to a system without specifying rules or constraints, that might be unknown at the time of training. In other words, a trained system is guaranteed to fail for a use case not specified at the time of design, while a learned system may *generalize* what it learned to correctly handle such cases. Learning in mechanical systems tends to be computationally easy, as the system employs a physical *learning rule* that modifies it due to use. This suggests that constructing large (many degrees of freedom) learning systems may be much more feasible than large designed systems. Moreover, continual learning of new tasks is an advantage that cannot be reproduced in design; a learning system may be able to adapt to new tasks without forgetting old ones, as we demonstrated on elastic networks.

Learning systems also have important disadvantages, both practical and conceptual. On a practical level, physically reasonable learning rules are quite a bit more limited than computer science machine learning rules. In some sense, they must be local in space and time, as a part of a system may only be modified according to the example it sees at that moment (with some leeway possible for neighbor interactions and temporal memory effects). Though such learning

rules naturally exist, e.g. Hebbian rule [152] in neurons, these rules likely heavily constrain the type of tasks physical systems can potentially learn to perform. In addition, engineering systems that implement such learning rules is far from trivial. Synthetic smart materials that change due to use are relatively new, and the extent of possible modifications is in active research. Thus, much more work would be required to construct synthetic physical systems that can learn any meaningful tasks. We do note that such research is currently actively undertaken for both self-folding origami and elastic networks.

A more fundamental disadvantage of learning is the issue of *interpretability* – the understanding of what the system has learned. Plaguing both neuroscience and machine learning, this idea that the system learned to perform a task does not mean that we, as its creators, understand how it actually does it. A learning system generically extracts some statistical information from the examples it observes, informing it on how to react if such correlations are again presented in the future. Unfortunately, the extracted information is generally opaque to both the trainer and the user. The problem is even worse when the learning system embodies a very complicated mathematical model (such as neural networks, natural and artificial). Our learning mechanical systems are somewhat similar. For example, the origami classifiers learn a suitable landscape for classifying the input forces, but what the learning rule actually optimizes is not completely clear (chapter 5), i.e. it is hard to tell what features are programmed into the energy landscape.

In this age, learning algorithms have taken off as a means of accomplishing complicated computational tasks, from detecting fraudulent interactions [169] to playing Go [170]. We propose to use these lessons in the creation of new classes of mechanical systems, possibly fit for accomplishing tasks previously impervious to design. A learning system may in some cases be trained by an end-user for specific tasks, allowing for more flexibility and usability than designed systems. Though methods of design will assuredly dominate the engineering process of most machines in the foreseeable future, learning physical machines might in fact be able to successfully occupy niches owing to some distinct advantages of learning.

Furthermore, we believe that mechanical systems themselves are an interesting avenue for

studying different concepts of learning. This work explored such ideas by identifying physically motivated learning rules, training origami and elastic networks by a user through analogies of both supervised and unsupervised learning. The migration of ideas from computer science to physics has often been be reciprocal [171], as exemplified by the recent connection of spin-glass physics with deep neural networks [172]. It is thus interesting to ask whether physically inspired systems can serve as a basis for constructing machine learning algorithms, perhaps endowed with potential novel properties. Finally, recent years saw massive interest in interpretable learning algorithms [173]. Mechanical systems have been used by society for so long, that people have developed strong intuitions about how these system operate.

Indeed, owing to model simplicity, our learning elastic networks are relatively easy to interpret in terms of the features that are encoded in the energy landscape during training. Moreover, these networks are naturally symmetric to translations and rotations (as all rigid bodies are), so that the learned solutions are symmetric to these operations. Enforcing symmetries on learning algorithms is a current research focus in machine learning [174]. Thus, we propose that physical, and especially mechanical systems, are particularly well suited for studying unique and interpretable learning algorithms. We hope that such efforts could enable future physics research to establish a general, physically motivated and interpretable theory of learning.

Appendix A

Technical Information

A.1 Origami modeling and folding

Large parts of this work, in particular chapters 2-4, were based on a specific mechanical system model: self-folding origami. We have studied these systems through several methods: simplified energy based models, realistic models with finite element simulations, and experiments. In this section we will discuss some technical details about these methods.

A.1.1 Simplified energy model

The theoretical model used to compute the energy of folded states in self-folding origami, and to numerically fold the sheet due to external forces, was explained in detail in previous chapters (e.g. Chapter 3, Appendix B). Here we briefly discuss several issues regarding the implementation of this model. The simplified energy model and numerical folding of origami were implemented as MATLAB codes, available upon request.

A fundamental issue in the study of origami is the definition of a folding mode. In principle, self-folding modes are $1d$ motions, completely defined by a discrete choice (between the exponentially many discrete modes), and one continuous number detailing the amount of overall folding

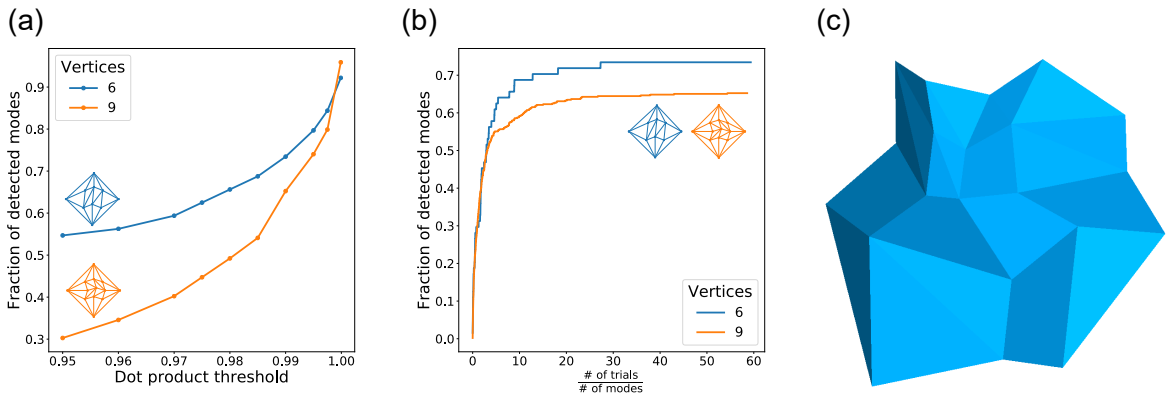

Figure A.1: Counting folding modes and angle verification
a) Relatively small changes in the dot product detection threshold significantly affects the number of detected modes, as many folding modes may be clustered close together. b) The number of detected modes saturates after a number of random trials equal ~ 20 times the number of expected folding modes $2^{\#\text{ vertices}}$ (data shown for dot product threshold of 0.95). c) Representative folded configurations are checked to ensure no sheet faces are colliding.

(e.g. ρ). However, for a generic pattern it is hard to find these $1d$ motions from the pattern geometry [81].

Thus, rather than computing the folding modes ab initio, we find them using constrained energy optimization. For a pattern with n creases, a folding vector $\boldsymbol{\rho}$ is sampled uniformly on a surface of the $(n-1)$-sphere of radius $\|\boldsymbol{\rho}\| \equiv \rho \sim 1$. This folded state is in general not part of a folding mode. We can find such folding modes by locally minimizing the energy $E(\boldsymbol{\rho})$ subject to the constraint of fixed folding magnitude ρ. For origami with uniform stiffness, these local minima $\boldsymbol{\rho}^*$ are guaranteed to be configurations belonging to a folding mode that emanates from the flat state $\boldsymbol{\rho} = 0$. Moreover, for small enough ρ these vectors are a good representation of the folding motion, as one can approximate it as $\boldsymbol{\rho}_{\text{mode}}(\rho) \approx \rho \cdot \boldsymbol{\rho}^*$. This constrained minimization is performed using MATLAB's fmincon function. After a mode $\boldsymbol{\rho}^*$ is found, we normalize it so that $\|\boldsymbol{\rho}^*\| = 1$.

Now that we can detect folding modes for an origami sheet, we wish to enumerate them. Numerical optimization is not exact and gives rise to numerical errors. Thus, if optimization starts at two separate but close initial vectors, it might converge on two slightly different points on the sphere, even if they both correspond to the same mode. To ensure that modes are not detected and counted twice, we employ a simple check on every new candidate mode: if the dot product

of it's representative vector $\boldsymbol{\rho}^*$ with all previous accepted mode is smaller than some threshold, the mode is accepted and added to the least of modes. If, on the other hand, the candidate mode has a high dot product with any previous mode, it is regarded as already detected and discarded. This simple algorithm generally results in mode under-counting, as some genuinely independent folding motions could be very similar, as seen in real origami models [175]. Fig. A.1a shows the modes counted after long sampling, as a function of the dot product threshold. The dot product threshold we employ varies somewhat for different tasks, but is generally in the range $0.95 - 0.99$.

Another issue that leads to mode under-counting is the sample size. If a pattern has N modes, it is clear that we need to sample at least N points on the sphere to have a chance of detecting all of them. Of course, many random samples lead to repeated modes, so in practice we have to sample many multiple times of N. Assuming the actual number of modes in a self-folding pattern V internal vertices is $N \approx 2^V$ [81], we find that the number of samples required to find at least 90% of the branches is around $20N$ (Fig. A.1b). Although both considerations lead to under-counting of the folding modes, choosing a high dot product threshold and large enough sampling, we can guarantee finding a representative fraction of the folding modes and thus establish the exponential relations described in chapter 3.

Finally, we note that numerically folding the origami is limited to relatively small angles, such that all dihedral angles are smaller than π. If a dihedral angle is equal to π, this means that on two faces of a physical sheet collide and further folding is impossible. Thus, our simplified energy model is completely inadequate for folded configurations with any $\rho \geq \pi$. In fact, even if no angle is that large, distant faces may still collide due to aggregated folding effects. To avoid such issues we generally look at relatively small overall folding, so that all dihedral angels are much smaller than π. In large patterns where colliding faces may be a concern, we use the MeshLab program to render $3d$ images of the folded sheet and visually check that there are no such collisions (Fig. A.1c).

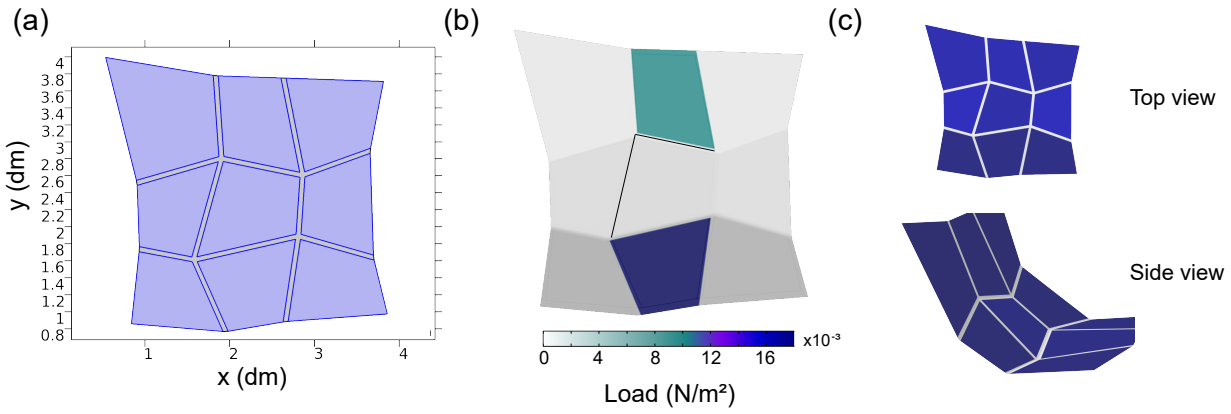

Figure A.2: Finite element simulation of origami sheets
a) Geometry of a self-folding sheet. The side of each face is $L \sim 10cm$ and the creases have a small width $w \sim 0.5cm$. b) In finite element simulations, sheets are folded by applying vertical loads on a few faces. Folding is then numerically performed until a stable equilibrium is reached. c) The folded structures are examined and compared to the folding modes predicted by our simplified energy model.

A.1.2 Finite element simulations

Our simplified energy models are derived to incorporate all crucial aspects of self-folding origami. Still, these are idealizations of real elastic thin sheets. To study how the complicating factors of realistic thin sheets modify our results, we simulate origami patterns with finite element methods (FEM) using COMSOL Multiphysics [176]. These simulations generally show that our simplified energy models capture the essential details of real origami.

To perform these simulations we use COMSOL's $2d$ plate model. The chosen geometry of the pattern is encoded, with the faces having a length scale of $L \sim 10cm$, and creases of width $w \sim 0.5cm$ (Fig. A.2a). The width of the sheet is chosen to be $d = 0.01cm$, both for faces and creases. Then, to facilitate folding mainly at the creases, we choose two different materials for faces and creases. The faces are chosen to have the properties of a relatively stiff material like Melamine resin or Acetal (with Young's modulus $Y = 6GPa$), while the creases are made of a much softer material like a Silicone elastomer (Young's modulus $Y = 5MPa$). These choices cause essentially all bending to occur at the creases, as dictated by the simplified energy model.

An important difference between the finite element simulation and the simplified model is in

the folding method. Here, we fold the pattern by applying area loads on some faces of the sheet, while clamping some other elements (points or lines) in place. This method is much closer to how origami would be folded in a real experiment. Fig. A.2b shows how vertical surface loads of different magnitudes are applied (upwards) on two of the faces, in order to fold the sheet. In this example, the sheet is constrained by requiring that two lines (highlighted in black) remain in place. The loads are applied, and the simulation folds the sheet until a force equilibrium is reached. Our simulations are generally computed with meshes built using the 'extremely fine' setting, and excluding geometric non-linearities.

The folded sheets are then examined carefully (Fig. A.2c). To compare these results with the folding modes predicted by our simplified model, we estimate the crease dihedral angles from these structures by sampling a few points on the faces. When compared to the simple model modes, we find a good agreement between the two sets of dihedral angles, with differences usually bound by 20%. Furthermore, the dihedral angle ranking in both models is usually the same, so that the folded structures do appear to be very similar. Other important elements of our simplified model, such as face bending localizing to a thin 'crease', and the idea of folding mode decided by mechanical advantage, were discussed in chapters 2 and 3. We conclude that a realistic and detailed finite element simulations of self-folding sheets are in agreement with the essential features of our simplified energy model. This observation grants us some confidence that the more interesting ideas we study in the simplified model, such as the methods of design or learning in these sheets, could be recreated in real sheets.

A.1.3 Experimental models

One type of experimental origami model is made by cutting 120 lb cardstock using a laser cutter. Patterns are roughly of size 10 cm x 10 cm. Creases were created using a perforation pattern of 0.6 mm cuts with 0.7 mm gaps. These patterns were folded by hand according to their respective designed MV configurations (Figure A.3).

A second type of experimental prototype uses two identical PVC sheets of width $\sim$ 0.5 cm, cut

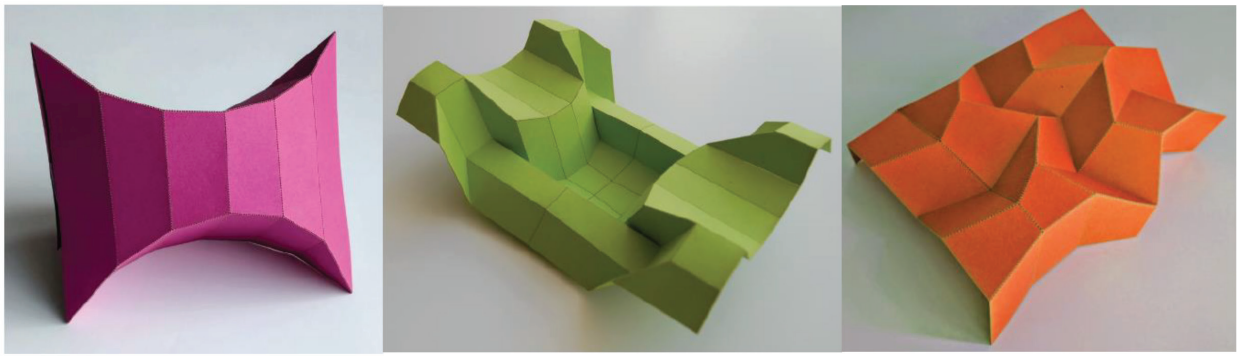

Figure A.3: Cardstock self-folding origami patterns

to the right pattern with a laser cutter. A sheet of paper is then sandwiched between the two PVC sheets, allowing for creases of width ~ 0.5 cm .

A.2 Energy landscapes for general factor graphs

Much of the work on self-folding origami, presented here, was accomplished using specialized Matlab code discussed earlier. To explore physical design and learning principles in a much broader context, we have developed a general purpose for simulation of generic conservative physical systems. This set of tools, developed in Cython-boosted Python, facilitates the simulation of both the thermodynamics and design\learning of any friction-less mechanical system. Fundamentally, we utilize the fact that any conservative physical system can be represented by a factor graph, giving rise to a huge class of generic energy landscapes. Specifically, we use these ideas to construct general elastic networks with linear or non-linear springs, and then explore such networks.

In this section, we elucidate the representation of conservative physical systems as factor graphs and describe how to construct the elastic networks of chapter 6 using these tools. We then discuss how these landscapes can be 'physically' explored, and how they can be modified by either design or learning processes. The Python codes for factor graph based energy landscapes are available on GitHub [177].

A.2.1 Physical systems as factor graphs

The static properties of any classical physical system, subjected only to conservative forces and fields, can be fully described by an energy function $E(\boldsymbol{x};\boldsymbol{p})$, where $\boldsymbol{x}$ stand for the configuration for all the physical degrees of freedom in the system, and $\boldsymbol{p}$ specifies all the interaction parameters. We define the configuration vector $\boldsymbol{x}$ as the physical state of the system. For example, in an elastic network, $\boldsymbol{x}$ corresponds to the location of all the nodes in the network, while in origami sheets, the state could be defined as the dihedral folding angles of all creases. We note that the physical state $\boldsymbol{x}$ is in general constructed of the physical description of multiple discrete 'objects' (e.g. node locations in the network).

Besides specifying the physical configuration of all the objects $\boldsymbol{x}$, computing the energy requires knowing how those discrete objects interact. This interaction is defined by the physical force these objects exert, whose description usually contains some parameters $\boldsymbol{p}$ that determine the exact form of the interaction. For example, the energy of two nodes in positions x_a, x_b (in one dimension), connected by a linear spring, is $E = \frac{1}{2}k(x_b - x_a)^2$. Here, we may set $\boldsymbol{x} = \{x_a, x_b\}$ and $\boldsymbol{p} = \{k\}$, such that the energy of the linear spring is defined as $E = f(\boldsymbol{x};\boldsymbol{p}) \equiv \frac{1}{2}p_1(x_2 - x_1)^2$. Pointless as this may seem for any particular single interaction, such a representation can be very powerful when multiple different interactions are considered between different types of discrete physical objects.

Note that in many typical examples in physics, the energy can be written as a sum over interaction energies of many interactions, each between a small set of physical objects. In an elastic network, one can write the total energy as a sum of spring energies, each spring connecting a pair of nodes. If these interactions are similar in form, yet different in the identity of interacting objects and the values of the interaction parameters, one can write the total energy as a sum of interaction terms

$$E_{\text{Total}}(\boldsymbol{x};\boldsymbol{p}) = \sum_{i \in \text{Interactions}} E_i(\boldsymbol{x}_i;\boldsymbol{p}_i). \tag{A.1}$$

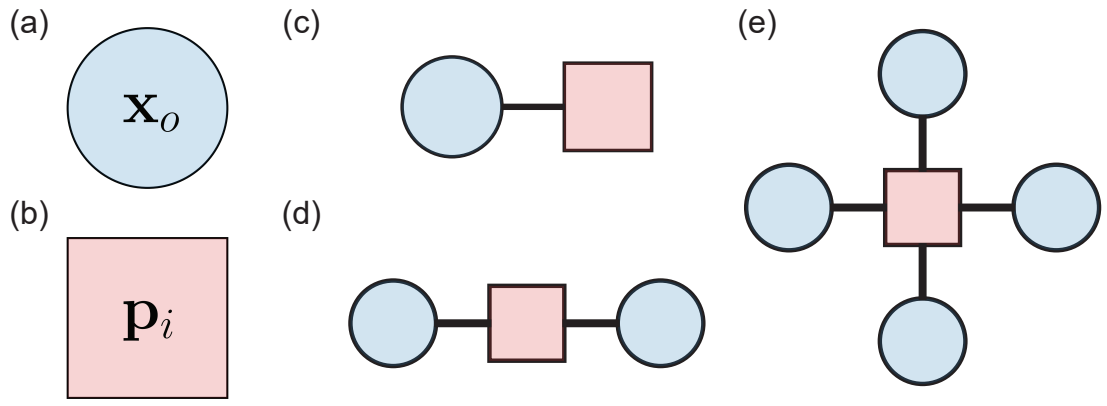

Figure A.4: Physical systems as factor graphs
a) Circle denotes a physical object. b) Square stands for an interaction term. c) Unary interaction of a physical object (e.g. external field). d) Binary interaction between two objects (e.g. spring connecting 2 nodes). e) Interaction connecting four physical variables (e.g. origami 4-vertex).

This sum can be represented graphically by a construction known as a factor graph, a bipartite graph with two types of elements, often drawn as circles and squares (Fig. A.4(a,b)). Let a physical object o be represented by a circle, associated with all the physical degrees of freedom $\boldsymbol{x}_o$ of the object. In addition, let an interaction i be represented by a square, associated with all the interaction parameters $\boldsymbol{p}_i$, and the functional form of the energy for that interaction $E_i(\boldsymbol{x}; \boldsymbol{p}_i)$. An interaction between different objects (or between objects and fields) can now be drawn as circles and squares that are connected by lines associating objects with their appropriate interactions. See Fig. A.4(c-e) for examples of graphical representations of different interaction types, including binary springs and origami 4-vertices. Note that in such factor graphs, lines always connect circles (objects) to squares (interactions). No line connects two circles or two squares, and all squares have to be connected to circles.

In general, there can exist multiple types of objects and interactions in the same system (Figure A.5). A simple example could be a system of Ising spins, for which there is one type of physical object (simple binary spins $x_o = \pm 1$), but there can be more than one interaction (nearest neighbor binary interaction, and interactions with an external field). Another example could be 3 nodes connected by 2 springs, and having an additional bond bending interaction. In this example the bond bending energy is effectively a 3-body interaction that is not symmetric in the physical

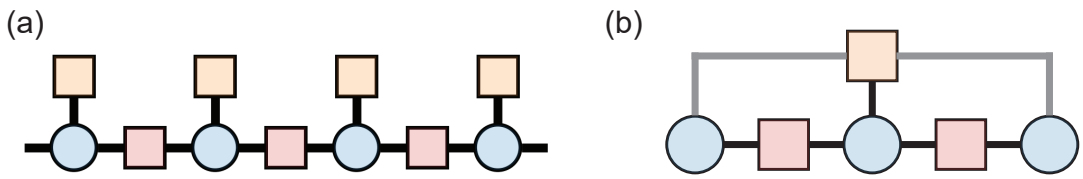

Figure A.5: More complex networks as factor graphs
a) Ising chain where spins interact with a local external field (orange squares) and their nearest neighbors (res squares). b) 3 nodes connected by 2 springs (red squares) and a bond bending energy term that treats different node locations differently (orange square).

objects (the locations of the 3 nodes enter differently in the bond bending energy computation). When an interaction is not symmetric in the objects associated with it, care should be taken to distinguish the lines connecting these objects and the interaction.

Large networks of interacting objects can be graphically represented as multiple object circles and interaction squares connected in a general bipartite graph. Given the full description of the physical degrees of freedom $\boldsymbol{x}$ (contents of each circle) and the interaction forms and parameters between them $\boldsymbol{p}$ (contents of each square), one can compute the energy of the system as a sum over all the squares in the diagram.

Crucially, this pictorial description of a physical system can readily be implemented as a computer code. We have developed Python software that constructs generic factor graphs, with generic descriptions of what the objects (circles) and interactions (squares) could be. These tools are thus able to compute and explore energy landscapes of diverse classes of systems, including elastic networks, origami, polymers, spin models, constraint satisfaction problems, electrostatics, and many more [177].

A.2.2 Elastic networks

The elastic networks studied in chapter 6 were constructed using the factor graphs described above. In these systems, the physical variables corresponds to node position, so that each circle in the factor graph corresponds to a single node. If the node is embedded in $d-$dimensional space, the circle contains a $d-$dimensional position vector. The interactions in the elastic network, i.e. the

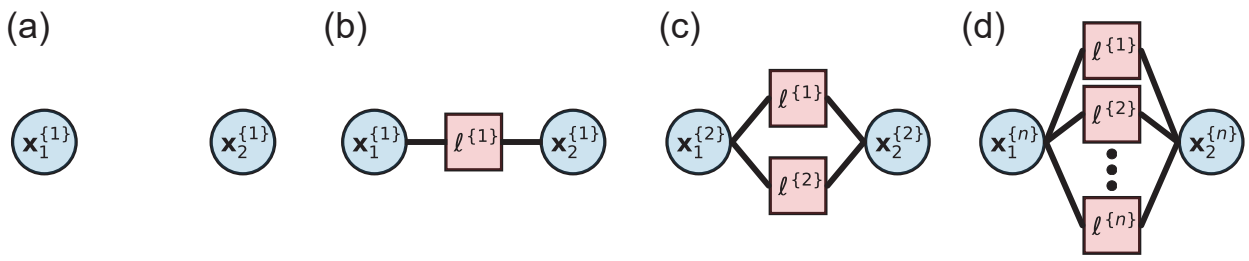

Figure A.6: Factor graphs of a growing elastic network
a) At first, nodes are not connected by any spring. b) When nodes are placed in position {1}, a spring grows between them with a rest length $\ell^{\{1\}}$ corresponding to their distance. c) As the nodes are placed displaced to position {2}, a new spring of appropriate rest length $\ell^{\{2\}}$ is formed. d) The spring growth process is repeated for all stored states, resulting in a factor graph with many squares (spring interactions) connected between the pairs of circles (physical nodes).

physical springs, are represented by squares, each square corresponding to a single spring. We consider nonlinear springs, each contributing an energy term (see Fig. 6.2):

$$E_{\text{spring}}(u) = k\sigma^{\xi} \cdot \frac{u^2}{(1+u^2)^{1-0.5\xi}}, \tag{A.2}$$

with $u \equiv (r-l)\sigma^{-1}$ and r the distance between the two connected nodes. Since this distance is a property of these nodes and not the springs, the parameters in Eq. A.2 associated with the spring itself are the spring stiffness k, its rest length l, the core radius σ and the power parameter ξ. Thus, every square in the factor graph contains a vector of the form (k, l, σ, ξ). In our study, we fix most of these parameters, so that all our non-linear springs have the same values of k, σ. In every particular learning network, we also fix the power parameter ξ for all springs. Thus, all springs in the network are the same, except for their rest length l, that is decided when the spring grows during the network's placement in a particular state.

The elastic network starts with free nodes and no springs. When it is placed in the first state a spring grows between each pair of nodes in some proximity. The rest length of that spring l_1 is equal to the distance between the nodes. The factor graph corresponding to this is a square connecting two circles. To encode a second state in the system, the positions of all the nodes are

changed (so that the vector components in each circle change). Then, a new non-linear spring grows between the pair of nodes, with a different rest length l_2. Graphically, this process corresponds to connecting a second square between the same two circles. As more states are encoded, extra springs grow to stabilize them. The factor graphs associated with continual learning (for just two nodes) is shown in Fig. A.6. The entire elastic network is represented by factor graphs of this kind between all pairs of nodes. The total energy in the elastic networks may be computed as a sum over all the squares in the graph, each of which contributes a term with the form of Eq. A.2.

A.2.3 Optimization, design and learning

The total energy associated with a factor graph, computed as a sum over all squares in the graph, is the energy of one particular physical state of the system. In other words, energy is computed for one particular setting of all the physical variables (circles) and interactions (squares). Naturally, physical systems may change their configuration according to some deterministic or stochastic equations of motion. A simple case we consider is overdamped dynamics at zero temperature. In such cases the system will change it's state by descending to a nearby state with a smaller energy. The exception to this rule is when the system reaches a local minimum in the energy landscape, and is then said to be stable. Such overdamped dynamics can be stated as a gradient descent optimization algorithm, where in every time step the system changes slightly in the direction where the energy drops most sharply. Thus, physical overdamped dynamics can be simulated by a step wise modification of all the circles in the factor graph to minimize the overall energy. Gradient descent optimization was implemented in our Python code for the purpose of these dynamics, and used to relax elastic networks into their nearby stable states.

As opposed to the physical dynamics which modify the circles in the factor graph, design and learning have to do with modifying the system parameters for a particular purpose. Thus, if we restrict ourselves to systems that do not change the number of degrees of freedom (i.e. fixed structure of circles), design and learning entail modification of the contents of the squares. In the simplest cases, we can live the structure of the squares fixed (i.e. same interaction type), and only

change the interaction parameters. For example, we can change the stiffness coefficient or rest length of a spring.

In the factor graph representation, a design could be phrased as an optimization of some design function over the contents of all squares. Such an optimization is a one step process, where the optimal solution is the optimal squares (i.e. interactions that best facilitate the design goal). This approach was used in Chapter 6 to find a design solution for linear springs that stabilize multiple desired states. In that example, the design function we optimized was the residual forces on the nodes due to the springs. When these forces vanish at the desired physical configuration, the system is in a stable state (or a saddle point).

A physical learning algorithm also modifies the squares of the factor graph, yet with some notable differences compared to design. Chiefly, the function optimized in learning is some kind of a cost function that always contains information about training examples (examples do not have to be incorporated into the design process). In physical systems, we also expects the training examples to be exhibited in sequence, so that training is a gradual process (in contrast to the one step nature of designed optimization). In our learned networks, the training examples are the sequence of states to be stabilized, each of which forms a new square between pairs of circles. Our Python library was constructed to facilitate simple incorporation of design and learning schemes for generic factor graphs.

References

[1] Don Norman. *The design of everyday things: Revised and expanded edition*. Basic books, 2013.

[2] Charles Darwin. *On the origin of species, 1859*. Routledge, 2004.

[3] John William S Pringle. On the parallel between learning and evolution. *Behaviour*, 1951.

[4] Pankaj Mehta, Marin Bukov, Ching-Hao Wang, Alexandre GR Day, Clint Richardson, Charles K Fisher, and David J Schwab. A high-bias, low-variance introduction to machine learning for physicists. *Physics Reports*, 2019.

[5] ML Maher. Machine learning in engineering design: Learning generalized design prototypes from examples. In *Development of Knowledge-Based Systems for Engineering*, pages 161–181. Springer, 1998.

[6] Murat Bengisu and Marinella Ferrara. *Materials that Move: Smart Materials, Intelligent Design*. Springer, 2018.

[7] Jason W Rocks, Henrik Ronellenfitsch, Andrea J Liu, Sidney R Nagel, and Eleni Katifori. Limits of multifunctionality in tunable networks. *Proceedings of the National Academy of Sciences*, 116(7):2506–2511, 2019.

[8] Nidhi Pashine, Daniel Hexner, Andrea J Liu, and Sidney R Nagel. Directed aging, memory and nature's greed. *arXiv preprint arXiv:1903.05776*, 2019.

[9] Koryo Miura. Method of packaging and deployment of large membranes. *Proc 31st Congr Int Astronaut Fed*, 1980.

[10] Edwin A Peraza-Hernandez, Darren J Hartl, Richard J Malak Jr, and Dimitris C Lagoudas. Origami-inspired active structures: a synthesis and review. *Smart Materials and Structures*, 23(9):094001, 2014.

[11] Christian D Santangelo. Extreme mechanics: Self-folding origami. *Annu. Rev. Conden. Ma. P.*, 8:165–183, 2017.

[12] Scott Waitukaitis, Rémi Menaut, Bryan Gin-ge Chen, and Martin van Hecke. Origami multistability: From single vertices to metasheets. *Physical Review Letters*, 114(5):055503, 2015.

[13] Jesse L Silverberg, Jun-Hee Na, Arthur A Evans, Bin Liu, Thomas C Hull, Christian D Santangelo, Robert J Lang, Ryan C Hayward, and Itai Cohen. Origami structures with a critical transition to bistability arising from hidden degrees of freedom. *Nat. Mater.*, 14(4):389–393, 1 April 2015.

[14] Levi H Dudte, Etienne Vouga, Tomohiro Tachi, and L Mahadevan. Programming curvature using origami tessellations. *Nat. Mater.*, 25 January 2016.

[15] Jason W Rocks, Nidhi Pashine, Irmgard Bischofberger, Carl P Goodrich, Andrea J Liu, and Sidney R Nagel. Designing allostery-inspired response in mechanical networks. *Proceedings of the National Academy of Sciences*, 114(10):2520–2525, 2017.

[16] Fl J Lockett and FJ Lockett. *Nonlinear viscoelastic solids*. Academic Press London, 1972.

[17] Katia Bertoldi, Vincenzo Vitelli, Johan Christensen, and Martin van Hecke. Flexible mechanical metamaterials. *Nat. Rev. Mater.*, 2(11):17066, 2017.

[18] Daniel R Reid, Nidhi Pashine, Justin M Wozniak, Heinrich M Jaeger, Andrea J Liu, Sidney R Nagel, and Juan J de Pablo. Auxetic metamaterials from disordered networks. *Proceedings of the National Academy of Sciences*, 115(7):E1384–E1390, 2018.

[19] Klaus Kroy. Elasticity, dynamics and relaxation in biopolymer networks. *Current opinion in colloid & interface science*, 11(1):56–64, 2006.

[20] Martin Goldstein. Viscous liquids and the glass transition: a potential energy barrier picture. *The Journal of Chemical Physics*, 51(9):3728–3739, 1969.

[21] Matthew B Pinson, Menachem Stern, Alexandra Carruthers Ferrero, Thomas A Witten, Elizabeth Chen, and Arvind Murugan. Self-folding origami at any energy scale. *Nat. Commun.*, 8:15477, 18 May 2017.

[22] Menachem Stern, Matthew B Pinson, and Arvind Murugan. The complexity of folding self-folding origami. *Physical Review X*, 7(4):041070, 2017.

[23] Menachem Stern, Viraaj Jayaram, and Arvind Murugan. Shaping the topology of folding pathways in mechanical systems. *Nature communications*, 9(1):4303, 2018.

[24] Menachem Stern, Chukwunonso Arinze, Leron Perez, Stephanie Palmer, and Arvind Murugan. Supervised learning in a mechanical system. *arXiv preprint arXiv:1910.09547*, 2019.

[25] Menachem Stern, Matthew B Pinson, and Arvind Murugan. Learned multi-stability in mechanical networks. *arXiv preprint arXiv:1902.08317*, 2019.

[26] Sergio Pellegrino. *Deployable Structures*. Springer, May 2014.

[27] Pedro M Reis, Heinrich M Jaeger, and Martin van Hecke. Designer matter: A perspective. *Extreme Mechanics Letters*, 5:25–29, December 2015.

[28] Howon Lee, Chunguang Xia, and Nicholas X Fang. First jump of microgel; actuation speed enhancement by elastic instability. *Soft Matter*, 6(18):4342–4345, 2010.

[29] Yoël Forterre, Jan M Skotheim, Jacques Dumais, and Lakshminarayanan Mahadevan. How the venus flytrap snaps. *Nature*, 433(7024):421–425, 2005.

[30] Jongmin Shim, Claude Perdigou, Elizabeth R Chen, Katia Bertoldi, and Pedro M Reis. Buckling-induced encapsulation of structured elastic shells under pressure. *Proc. Natl. Acad. Sci. U.S.A.*, 109(16):5978–5983, April 2012.

[31] Tomohiro Tachi. One-dof cylindrical deployable structures with rigid quadrilateral panels. *Evolution and Trends in Design, Analysis and Construction of Shell and Spatial Structures: Proceedings*, 2010.

[32] Charles Hoberman. Reversibly expandable doubly-curved truss structure. US Patent Office, July 1990.

[33] Jean-Pierre Merlet. *Parallel Robots*. Springer Science & Business Media, December 2012.

[34] Erik D Demaine and Joseph O'Rourke. *Geometric Folding Algorithms*. Linkages, Origami, Polyhedra. Cambridge University Press, July 2007.

[35] Andrea J Liu and Sidney R Nagel. The jamming transition and the marginally jammed solid. *Annual Review of Condensed Matter Physics*, 1(1):347–369, August 2010.

[36] Tomohiro Tachi. Generalization of rigid foldable quadrilateral mesh origami. *Proceedings of the International Association for Shell and Spatial Structures (IASS) Symposium*, 2009.

[37] Tomohiro Tachi. Geometric considerations for the design of rigid origami structures. In *Proceedings of the International Association for Shell and Spatial Structures (IASS) Symposium*, volume 12, pages 458–460, 2010.

[38] Yan Chen, Rui Peng, and Zhong You. Origami of thick panels. *Science*, 349(6246):396–400, July 2015.

[39] Arthur A Evans, Jesse L Silverberg, and Christian D Santangelo. Lattice mechanics of origami tessellations. *Physical Review E*, 92(1):013205, 2015.

[40] Tomohiro Tachi. Simulation of rigid origami. *Origami*, 4:175–187, 2009.

[41] David A Huffman. Curvature and creases: A primer on paper. *IEEE Trans. Comput.*, C-25(10):1010–1019, 1 October 1976.

[42] Sarah-Marie Belcastro and Thomas C Hull. Modelling the folding of paper into three dimensions using affine transformations. *Linear Algebra and its applications*, 348(1):273–282, 2002.

[43] Tomohiro Tachi. Design of infinitesimally and finitely flexible origami based on reciprocal figures. *Journal for Geometry and Graphics*, 16(2):223–234, 2012.

[44] Weina Wu and Zhong You. Modelling rigid origami with quaternions and dual quaternions. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 466, pages 2155–2174. The Royal Society, 2010.

[45] Jun-Hee Na, Arthur A. Evans, Jinhye Bae, Maria C. Chiappelli, Christian D. Santangelo, Robert J. Lang, Thomas C. Hull, and Ryan C. Hayward. Programming reversibly self-folding origami with micropatterned photo-crosslinkable polymer trilayers. *Advanced Materials*, 27(1):79–85, 2015.

[46] Marshall Bern and Barry Hayes. The complexity of flat origami. *Proceedings of the 7th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 175–183, 1996.

[47] Samuel M Felton, Michael T Tolley, Byunghyun Shin, Cagdas D Onal, Erik D Demaine, Daniela Rus, and Robert J Wood. Self-folding with shape memory composites. *Soft Matter*, 9(32):7688–7694, 24 July 2013.

[48] Elliot Hawkes, Byoungkwon An, Nadia M Benbernou, Hiroto Tanaka, Sangbae Kim, Erik D Demaine, Daniela Rus, and Robert J Wood. Programmable matter by folding. *Proceedings of the National Academy of Sciences*, 107(28):12441–12445, 13 July 2010.

[49] Kaori Kuribayashi, Koichi Tsuchiya, Zhong You, Dacian Tomus, Minoru Umemoto, Takahiro Ito, and Masahiro Sasaki. Self-deployable origami stent grafts as a biomedical application of ni-rich TiNi shape memory alloy foil. *Materials Science and Engineering: A*, 419(1–2):131–137, 15 March 2006.

[50] Tae Soup Shim, Shin-Hyun Kim, Chul-Joon Heo, Hwan Chul Jeon, and Seung-Man Yang. Controlled origami folding of hydrogel bilayers with sustained reversibility for robust microcarriers. *Angew. Chem. Int. Ed Engl.*, 51(6):1420–1423, 6 February 2012.

[51] Robert J Lang. The science of origami. *Physics world*, 20(2):30, 2007.

[52] Thomas Hull. *Project Origami*. Activities for Exploring Mathematics, Second Edition. CRC Press, December 2012.

[53] Bryan Gin-ge Chen, Bin Liu, Arthur A Evans, Jayson Paulose, Itai Cohen, Vincenzo Vitelli, and Christian D Santangelo. Topological mechanics of origami and kirigami. *Phys. Rev. Lett.*, 116(13):135501, March 2016.

[54] Alex Lobkovsky, Sharon Gentges, Hao Li, David Morse, and Thomas A Witten. Scaling properties of stretching ridges in a crumpled elastic sheet. *Science*, 270(5241):1482, 1995.

[55] Thomas A Witten. Stress focusing in elastic sheets. *Reviews of Modern Physics*, 79(2):643, 2007.

[56] José M Zanardi Ocampo, Pablo O Vaccaro, Thomas Fleischmann, Te-Sheng Wang, Kazuyoshi Kubota, Tahito Aida, Toshiaki Ohnishi, Akira Sugimura, Ryo Izumoto, Makoto Hosoda, and Shigeki Nashima. Optical actuation of micromirrors fabricated by the micro-origami technique. *Appl. Phys. Lett.*, 28 October 2003.

[57] Yacov Kantor and David R Nelson. Crumpling transition in polymerized membranes. *Physical review letters*, 58(26):2774, 1987.

[58] Mark Bowick, Philippe Di Francesco, Oliver Golinelli, and Emmanuel Guitter. Three-dimensional folding of the triangular lattice. *Nuclear Physics B*, 450(3):463–494, 1995.

[59] Francois David and Emmanuel Guitter. Crumpling transition in elastic membranes: renormalization group treatment. *EPL (Europhysics Letters)*, 5(8):709, 1988.

[60] Jesse L Silverberg, Arthur A Evans, Lauren McLeod, Ryan C Hayward, Thomas Hull, Christian D Santangelo, and Itai Cohen. Using origami design principles to fold reprogrammable mechanical metamaterials. *Science*, 345(6197):647–650, 2014.

[61] Sahand Hormoz and Michael P Brenner. Design principles for self-assembly with short-range interactions. *Proceedings of the National Academy of Sciences*, 108(13):5193–5198, 29 March 2011.

[62] Vijay S Pande, Alexander Yu Grosberg, and Toyoichi Tanaka. Heteropolymer freezing and design: Towards physical models of protein folding. *Rev. Mod. Phys.*, 72(1):259–314, 1 January 2000.

[63] William M Jacobs, Aleks Reinhardt, and Daan Frenkel. Communication: Theoretical prediction of free-energy landscapes for complex self-assembly. *J. Chem. Phys.*, 142(2):021101, 14 January 2015.

[64] Zorana Zeravcic, Vinothan N Manoharan, and Michael P Brenner. Size limits of self-assembled colloidal structures made using specific interactions. *Proceedings of the National Academy of Sciences*, 111(45):15918–15923, 2014.

[65] Arvind Murugan, James Zou, and Michael P Brenner. Undesired usage and the robust self-assembly of heterogeneous structures. *Nat. Commun.*, 6:6203, 11 February 2015.

[66] Victor I Abkevich, Alexander M Gutin, and Eugene I Shakhnovich. Free energy landscape for protein folding kinetics: Intermediates, traps, and multiple pathways in theory and lattice model simulations. *J. Chem. Phys.*, 101(7):6052–6062, 1 January 1994.

[67] Martin Karplus. The levinthal paradox: yesterday and today. *Fold. Des.*, 2(4):S69–75, 1997.

[68] Cyrus Levinthal. How to fold graciously. *Mossbauer spectroscopy in biological systems*, 67:22–24, 1969.

[69] Pierluigi Crescenzi, Deborah Goldman, Christos Papadimitriou, Antonio Piccolboni, and Mihalis Yannakakis. On the complexity of protein folding. *J. Comput. Biol.*, 5(3):423–465, 1998.

[70] J Thomas Ngo, Joe Marks, and Martin Karplus. Computational complexity, protein structure prediction, and the levinthal paradox. In Kenneth M Merz, Jr. and Scott M Le Grand, editors, *The Protein Folding Problem and Tertiary Structure Prediction*, pages 433–506. Birkhäuser Boston, 1994.

[71] Giulio Biroli, Simona Cocco, and Rémi Monasson. Phase transitions and complexity in computer science: an overview of the statistical physics approach to the random satisfiability problem. *Physica A: Statistical Mechanics and its Applications*, 306:381–394, 1 January 2002.

[72] Zachary Abel, Jason Cantarella, Erik D Demaine, David Eppstein, Thomas C Hull, Jason S Ku, Robert J Lang, and Tomohiro Tachi. Rigid origami vertices: Conditions and forcing sets. *arXiv*, 6 July 2015.

[73] Brad Ballinger, Mirela Damian, David Eppstein, Robin Flatland, Jessica Ginepro, and Thomas Hull. Minimum forcing sets for miura folding patterns. In *Proceedings of the Twenty-sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '15, pages 136–147, Philadelphia, PA, USA, 2015. Society for Industrial and Applied Mathematics.

[74] Esther M Arkin, Michael A Bender, Erik D Demaine, Martin L Demaine, Joseph SB Mitchell, Saurabh Sethia, and Steven S Skiena. When can you fold a map? *Computational Geometry*, 29(1):23–46, 2004.

[75] Shivendra Pandey, Margaret Ewing, Andrew Kunas, Nghi Nguyen, David H Gracias, and Govind Menon. Algorithmic design of self-folding polyhedra. *Proceedings of the National Academy of Sciences*, 108(50):19885–19890, 1 January 2011.

[76] Byoungkwon An, Nadia Benbernou, Erik D Demaine, and Daniela Rus. Planning to fold multiple objects from a single self-folding sheet. *Robotica*, 29(1):87–102, 1 January 2011.

[77] Tomohiro Tachi and Thomas C Hull. Self-foldability of rigid origami. *J. Mech. Robot.*, 9(2):021008, 2017.

[78] Toshikazu Kawasaki. On the relation between mountain-creases and valley-creases of a flat origami. In *Proceedings of the 1st International Meeting of Origami Science and Technology*, pages 229–237, 1989.

[79] Thomas Hull. On the mathematics of flat origamis. *Congressus numerantium*, pages 215–224, 1994.

[80] Kurt Binder and Peter A Young. Spin glasses: Experimental facts, theoretical concepts, and open questions. *Reviews of Modern physics*, 58(4):801, 1986.

[81] Bryan Gin-ge Chen and Christian D Santangelo. Branches of triangulated origami near the unfolded state. *Phys. Rev. X*, 8(1):011034, 2018.

[82] Yan V Fyodorov. Complexity of random energy landscapes, glass transition, and absolute value of the spectral determinant of random matrices. *Phys. Rev. Lett.*, 92(24):240601, 18 June 2004.

[83] Bonnie Berger and Tom Leighton. Protein folding in the hydrophobic-hydrophilic (HP) model is NP-complete. *J. Comput. Biol.*, 5(1):27–40, 1998.

[84] Florent Krzkakala and Lenka Zdeborová. Phase transitions and computational difficulty in random constraint satisfaction problems. *J. Phys. Conf. Ser.*, 95(1):012012, 2008.

[85] Inês Lynce and Joël Ouaknine. Sudoku as a SAT problem. In *In Proc. of the Ninth International Symposium on Artificial Intelligence and Mathematics*, 2006.

[86] Omer Gottesman, Efi Efrati, and Shmuel M Rubinstein. Furrows in the wake of propagating d-cones. *Nature Communications*, 6, 2015.

[87] A Jamie Wood. Witten's lectures on crumpling. *Physica A: Statistical Mechanics and its Applications*, 313(1):83–109, 2002.

[88] Vijay V Vazirani. *Approximation algorithms*. Springer Science & Business Media, 2013.

[89] Bin Liu, Jesse L Silverberg, Arthur A Evans, Christian D Santangelo, Robert J Lang, Thomas C Hull, and Itai Cohen. Topological kinematics of origami metamaterials. *Nat. Phys.*, 14:811–815, 2018.

[90] Charles W Wampler. Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods. *IEEE T. Syst. Man Cyb.*, 16(1):93–101, 1986.

[91] Charles W Wampler, Jonathan D Hauenstein, and Andrew J Sommese. Mechanism mobility and a local dimension test. *Mech. Mach. Theory*, 46(9):1193–1206, 2011.

[92] Naomi Oppenheimer and Thomas A Witten. Shapeable sheet without plastic deformation. *Phys. Rev. E*, 92(5):052401, 2015.

[93] Bastiaan Florijn, Corentin Coulais, and Martin van Hecke. Programmable mechanical metamaterials. *Phys. Rev. Lett.*, 113(17):175503, 2014.

[94] Yan Chen and Woon Huei Chai. Bifurcation of a special line and plane symmetric bricard linkage. *Mech. Mach. Theory*, 46(4):515–533, 2011.

[95] David Rocklin, Vincenzo Vitelli, and Xiaoming Mao. Folding mechanisms at finite temperature. *Preprint at http://arXiv.org/abs/1802.02704*, 2018.

[96] David H Myszka, Andrew P Murray, and Charles W Wampler. Mechanism branches, turning curves, and critical points. In *ASME 2012 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pages 1513–1525. American Society of Mechanical Engineers, 2012.

[97] David H Myszka, Andrew P Murray, and Charles W Wampler. Computing the branches, singularity trace, and critical points of single Degree-of-Freedom, Closed-Loop linkages. *J. Mech. Robot.*, 6(1):011006, 2013.

[98] Jon Kieffer. Differential analysis of bifurcations and isolated singularities for robots and mechanisms. *IEEE T. Robotic. Autom.*, 10(1):1–10, 1994.

[99] Eric J Deeds, Orr Ashenberg, Jaline Gerardin, and Eugene I Shakhnovich. Robust protein protein interactions in crowded cellular environments. *Proc. Natl. Acad. Sci. U. S. A.*, 104(38):14952–14957, 2007.

[100] William M Jacobs and Eugene I Shakhnovich. Evidence of evolutionary selection for cotranslational folding. *Proc. Natl. Acad. Sci. U. S. A.*, 114(43):11434–11439, 2017.

[101] Christopher M Dobson. Protein folding and misfolding. *Nature*, 426(6968):884–890, 2003.

[102] Richard Scheunemann Hartenberg and Jacques Denavit. *Kinematic synthesis of linkages*. McGraw-Hill, 1964.

[103] J Michael McCarthy. *Geometric design of linkages*, volume 11. Springer Science & Business Media, 2006.

[104] Bryan Gin-ge Chen, Nitin Upadhyaya, and Vincenzo Vitelli. Nonlinear conduction via solitons in a topological mechanical insulator. *Proc. Natl. Acad. Sci. U. S. A.*, 111(36):13004–13009, 2014.

[105] Larry L Howell. *Compliant mechanisms*. John Wiley & Sons, 2001.

[106] Sridhar Kota and GK Ananthasuresh. Designing compliant mechanisms. *Mech. Eng. CIME*, 117(11):93–97, 1995.

[107] Vasek Chvatal, Vaclav Chvatal, et al. *Linear programming*. Macmillan, 1983.

[108] Robert J Vanderbei et al. *Linear programming*. Springer, 2015.

[109] Douglas P Holmes, Matthieu Roché, Tarun Sinha, and Howard A Stone. Bending and twisting of soft materials by non-homogenous swelling. *Soft Matter*, 7(11):5188–5193, 2011.

[110] Eric Brown, Nicole A Forman, Carlos S Orellana, Hanjun Zhang, Benjamin W Maynor, Douglas E Betts, Joseph M DeSimone, and Heinrich M Jaeger. Generality of shear thickening in dense suspensions. *Nat. Mater.*, 9(3):220–224, 2010.

[111] Neil YC Lin, Christopher Ness, Michael E Cates, Jin Sun, and Itai Cohen. Tunable shear thickening in suspensions. *Proc. Natl. Acad. Sci. U. S. A.*, 113(39):10774–10778, 2016.

[112] Lisa M Nash, Dustin Kleckner, Alismari Read, Vincenzo Vitelli, Ari M Turner, and William TM Irvine. Topological mechanics of gyroscopic metamaterials. *Proceedings of the National Academy of Sciences*, 112(47):14495–14500, 2015.

[113] Jason Z Kim, Zhixin Lu, Steven H Strogatz, and Danielle S Bassett. Conformational control of mechanical networks. *Nature Physics*, page 1, 2019.

[114] Jason Z Kim, Zhixin Lu, and Danielle S Bassett. Design of large sequential conformational change in mechanical networks. *arXiv preprint arXiv:1906.08400*, 2019.

[115] Stephen Grossberg. Adaptive pattern classification and universal recoding: Ii. feedback, expectation, olfaction, illusions. *Biological cybernetics*, 23(4):187–202, 1976.

[116] Andrew Adamatzky. *Physarum machines: computers from slime mould*, volume 74. World Scientific, 2010.

[117] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[118] Leonard Mullins. Softening of rubber by deformation. *Rubber chemistry and technology*, 42(1):339–362, 1969.

[119] H_E Read and GA Hegemier. Strain softening of rock, soil and concretea review article. *Mechanics of Materials*, 3(4):271–294, 1984.

[120] AV Lyulin, B Vorselaars, MA Mazo, NK Balabaev, and MAJ Michels. Strain softening and hardening of amorphous polymers: Atomistic simulation of bulk mechanics and local dynamics. *EPL (Europhysics Letters)*, 71(4):618, 2005.

[121] Jan A Åström, PB Sunil Kumar, Ilpo Vattulainen, and Mikko Karttunen. Strain hardening, avalanches, and strain softening in dense cross-linked actin networks. *Physical Review E*, 77(5):051913, 2008.

[122] YW Deng, TL Yu, and CH Ho. Effect of aging under strain on the physical properties of polyester–urethane elastomer. *Polymer journal*, 26(12):1368, 1994.

[123] Seong-Gu Hong, Keum-Oh Lee, and Soon-Bok Lee. Dynamic strain aging effect on the fatigue resistance of type 316l stainless steel. *International Journal of Fatigue*, 27(10-12):1420–1424, 2005.

[124] Shan Zi Gui and Yukuo Nanzai. Aging in quenched poly (methyl methacrylate) under inelastic tensile strain. *Polymer journal*, 33(5):444, 2001.

[125] Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.

[126] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.

[127] Anil K Jain, Robert P. W. Duin, and Jianchang Mao. Statistical pattern recognition: A review. *IEEE Transactions on pattern analysis and machine intelligence*, 22(1):4–37, 2000.

[128] Izabela K Kuder, Andres F Arrieta, Wolfram E Raither, and Paolo Ermanni. Variable stiffness material and structural concepts for morphing applications. *Progress in Aerospace Sciences*, 63:33–55, 2013.

[129] Geoff McKnight and Chris Henry. Variable stiffness materials for reconfigurable surface applications. In *Smart Structures and Materials 2005: Active Materials: Behavior and Mechanics*, volume 5761, pages 119–126. International Society for Optics and Photonics, 2005.

[130] Patrick T Mather, Xiaofan Luo, and Ingrid A Rousseau. Shape memory polymer research. *Annual Review of Materials Research*, 39:445–471, 2009.

[131] William B Cross, Anthony H Kariotis, and Frederick J Stimler. Nitinol characterization study. *NASA*, CR-1433, 1969.

[132] Michael Philen, Ying Shan, Kon-Well Wang, Charles Bakis, and Christopher Rahn. Fluidic flexible matrix composites for the tailoring of variable stiffness adaptive structures. In *48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 1703, 2007.

[133] Jing Jin, Shuangjun Chen, and Jun Zhang. Uv aging behaviour of ethylene-vinyl acetate copolymers (eva) with different vinyl acetate contents. *Polymer degradation and stability*, 95(5):725–732, 2010.

[134] A Boubakri, Nader Haddar, K Elleuch, and Yves Bienvenu. Impact of aging conditions on mechanical properties of thermoplastic polyurethane. *Materials & Design*, 31(9):4194–4201, 2010.

[135] Dimitris C Lagoudas. *Shape memory alloys: modeling and engineering applications*. Springer, 2008.

[136] Nathan C Keim, Joseph D Paulsen, Zorana Zeravcic, Srikanth Sastry, and Sidney R Nagel. Memory formation in matter. *Reviews of Modern Physics*, 91(3):035002, 2019.

[137] Jamie K Paik, An Byoungkwon, Daniela Rus, and Robert J Wood. Robotic origamis: Self-morphing modular robot. In *ICMC*, 2012.

[138] Johannes TB Overvelde, Twan A De Jong, Yanina Shevchenko, Sergio A Becerra, George M Whitesides, James C Weaver, Chuck Hoberman, and Katia Bertoldi. A three-dimensional actuated origami-inspired transformable metamaterial with multiple degrees of freedom. *Nature communications*, 7:10929, 2016.

[139] Sicong Shan, Sung H Kang, Jordan R Raney, Pai Wang, Lichen Fang, Francisco Candido, Jennifer A Lewis, and Katia Bertoldi. Multistable architected materials for trapping elastic strain energy. *Advanced Materials*, 27(29):4296–4301, 2015.

[140] Gabi Steinbach, Dennis Nissen, Manfred Albrecht, Ekaterina V Novak, Pedro A Sánchez, Sofia S Kantorovich, Sibylle Gemming, and Artur Erbe. Bistable self-assembly in homogeneous colloidal systems for flexible modular architectures. *Soft Matter*, 12(10):2737–2743, 2016.

[141] Lingling Wu, Xiaoqing Xi, Bo Li, and Ji Zhou. Multi-Stable mechanical structural materials. *Adv. Eng. Mater.*, 20(2):1700599, February 2018.

[142] Yi Yang, Marcelo A Dias, and Douglas P Holmes. Multistable kirigami for tunable architected materials. *Phys. Rev. Materials*, 2(11):110601, November 2018.

[143] Kaikai Che, Chao Yuan, Jiangtao Wu, H Jerry Qi, and Julien Meaud. Three-dimensional-printed multistable mechanical metamaterials with a deterministic deformation sequence. *Journal of Applied Mechanics*, 84(1):011004, 2017.

[144] Hang Yang and Li Ma. Multi-stable mechanical metamaterials with shape-reconfiguration and zero poisson's ratio. *Mater. Des.*, 152:181–190, August 2018.

[145] Fu et. al. Morphable 3d mesostructures and microelectronic devices by multistable buckling mechanics. *Nat. Mater.*, 17(3):268–276, March 2018.

[146] Daniel Hexner, Andrea J Liu, and Sidney R Nagel. Role of local response in manipulating the elastic properties of disordered solids by bond removal. *Soft matter*, 14(2):312–318, 2018.

[147] Henry Hess and Jennifer L Ross. Non-equilibrium assembly of microtubules: from molecules to autonomous chemical robots. *Chem. Soc. Rev.*, 46(18):5570–5587, September 2017.

[148] Abdul M Mohammed and Rebecca Schulman. Directing self-assembly of dna nanotubes using programmable seeds. *Nano letters*, 13(9):4006–4013, 2013.

[149] Midori Isobe and Ko Okumura. Initial rigid response and softening transition of highly stretchable kirigami sheet materials. *Scientific reports*, 6:24758, 2016.

[150] Jiayao Ma, Jichao Song, and Yan Chen. An origami-inspired structure with graded stiffness. *International Journal of Mechanical Sciences*, 136:134–142, 2018.

[151] Marileen Dogterom and Thomas Surrey. Microtubule organization in vitro. *Curr. Opin. Cell Biol.*, 25(1):23–29, February 2013.

[152] Donald O Hebb. *The organization of behavior*. Wiley, 1949.

[153] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[154] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

[155] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical learning with sparsity: the lasso and generalizations*. CRC press, 2015.

[156] John Hertz, Anders Krogh, and Richard G Palmer. *Introduction to the theory of neural computation*. Addison-Wesley/Addison Wesley Longman, 1991.

[157] Daniel J Amit, Hanoch Gutfreund, and Haim Sompolinsky. Storing infinite numbers of patterns in a spin-glass model of neural networks. *Physical Review Letters*, 55(14):1530, 1985.

[158] Daniel J Amit, Hanoch Gutfreund, and Haim Sompolinsky. Spin-glass models of neural networks. *Physical Review A*, 32(2):1007, 1985.

[159] Jerome H Friedman. Fast sparse regression and classification. *International Journal of Forecasting*, 28(3):722–738, 2012.

[160] Angshul Majumdar and Rabab K Ward. Non-convex group sparsity: Application to color imaging. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 469–472. IEEE, 2010.

[161] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2, 2010.

[162] Sinan Li, Pingang He, Jianhua Dong, Zhixin Guo, and Liming Dai. DNA-directed self-assembling of carbon nanotubes. *J. Am. Chem. Soc.*, 127(1):14–15, January 2005.

[163] Jeffrey D Hartgerink, Juan R Granja, Ronald A Milligan, and M Reza Ghadiri. Self-Assembling peptide nanotubes. *J. Am. Chem. Soc.*, 118(1):43–50, January 1996.

[164] Werner J Blau and Alexander J Fleming. Materials science. designer nanotubes by molecular self-assembly. *Science*, 304(5676):1457–1458, June 2004.

[165] Adel Javanmard and Andrea Montanari. Localization from incomplete noisy distance measurements. *Foundations of Computational Mathematics*, 13(3):297–345, 2013.

[166] John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[167] Elizabeth Gardner. The space of interactions in neural network models. *Journal of physics A: Mathematical and general*, 21(1):257, 1988.

[168] James C Maxwell. L. on the calculation of the equilibrium and stiffness of frames. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 27(182):294–299, 1864.

[169] Philip K Chan and Salvatore J Stolfo. Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In *KDD*, volume 1998, pages 164–168, 1998.

[170] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[171] Giuseppe Carleo, Ignacio Cirac, Kyle Cranmer, Laurent Daudet, Maria Schuld, Naftali Tishby, Leslie Vogt-Maranto, and Lenka Zdeborová. Machine learning and the physical sciences. *arXiv preprint arXiv:1903.10563*, 2019.

[172] Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics*, pages 192–204, 2015.

[173] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

[174] Robert Gens and Pedro M Domingos. Deep symmetry networks. In *Advances in neural information processing systems*, pages 2537–2545, 2014.

[175] Yan Chen, Huijuan Feng, Jiayao Ma, Rui Peng, and Zhong You. Symmetric waterbomb origami. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 472(2190):20150846, 2016.

[176] COMSOL AB. Comsol multiphysics v. 5.4. www.comsol.com. comsol ab, stockholm, sweden.

[177] Menachem Stern. FactorGraph-Landscapes, https://doi.org/10.5281/zenodo.3266234, July 2019.