



University of Pennsylvania  
**ScholarlyCommons**

---

Publicly Accessible Penn Dissertations


---

2022

## Complex Systems Engineering: Designing Advanced Functions In Dynamical And Mechanical Systems

Jinsu Kim  
*University of Pennsylvania*

Follow this and additional works at: <https://repository.upenn.edu/edissertations>

 Part of the [Engineering Mechanics Commons](#), and the [Neuroscience and Neurobiology Commons](#)

---

### Recommended Citation

Kim, Jinsu, "Complex Systems Engineering: Designing Advanced Functions In Dynamical And Mechanical Systems" (2022). *Publicly Accessible Penn Dissertations*. 5512.  
<https://repository.upenn.edu/edissertations/5512>

This paper is posted at ScholarlyCommons. <https://repository.upenn.edu/edissertations/5512>  
For more information, please contact [repository@pobox.upenn.edu](mailto:repository@pobox.upenn.edu).

---

# Complex Systems Engineering: Designing Advanced Functions In Dynamical And Mechanical Systems

## Abstract

From computation in neural networks to allostery in proteins, numerous natural and artificial systems are comprised of many interacting parts that give rise to advanced functions. To study such complex systems, a diverse array of interdisciplinary tools have been developed that relate the interactions of existing systems to their functions. However, engineering the interactions to perform designed functions in novel systems remains a significant challenge due to the nonlinearities in the interactions and the vast dimensionality of the design space. Here we develop design principles for complex dynamical and mechanical systems at the lowest level of their microstate interactions. In dynamical neural systems, we use methods from control theory and dynamical systems theory to mathematically map precise patterns of neural connectivity to the control of neural states in human and non-human brains (Chapter 2) and to the learning of computations on internal representations in artificial recurrent neural networks (Chapter 4). In mechanical systems, we use methods from algebraic geometry and dynamical systems to mathematically map precise patterns of mechanical constraints to design shape changes as a minimal model of protein allostery and cooperativity (Chapter 6) and to engineer mechanical metamaterials that possess arbitrarily complex shape changes (Chapter 8). These intuitive maps allow us to navigate previously unexplored design spaces in nonlinear and high-dimensional regimes, enabling us to reverse engineer form from function in novel complex systems that have yet to exist.

## Degree Type

Dissertation

## Degree Name

Doctor of Philosophy (PhD)

## Graduate Group

Bioengineering

## First Advisor

Danielle S. Bassett

## Keywords

complex systems, deep learning, distributed intelligence, dynamical systems, metamaterials, neural networks

## Subject Categories

Engineering Mechanics | Neuroscience and Neurobiology



COMPLEX SYSTEMS ENGINEERING:  
DESIGNING ADVANCED FUNCTIONS IN DYNAMICAL AND MECHANICAL  
SYSTEMS

Jinsu Kim

A DISSERTATION

in

Bioengineering

Presented to the Faculties of the University of Pennsylvania

in

Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

2022

Supervisor of Dissertation

---

Dani S. Bassett, J. Peter Skirkanich Professor of Bioengineering

Graduate Group Chairperson

---

Yale E. Cohen, Professor of Otorhinolaryngology

Dissertation Committee

David Issadore, Associate Professor of Bioengineering

George J. Pappas, UPS Foundation Professor of Electrical and Systems Engineering

Erol Akcay, Associate Professor of Biology

COMPLEX SYSTEMS ENGINEERING:  
DESIGNING ADVANCED FUNCTIONS IN DYNAMICAL AND MECHANICAL  
SYSTEMS

© COPYRIGHT

2022

Jinsu Kim

This work is licensed under the  
Creative Commons Attribution  
NonCommercial-ShareAlike 4.0  
License

To view a copy of this license, visit

<http://creativecommons.org/licenses/by-nc-sa/4.0/>

## DEDICATION

To the original Complex Systems Engineer,  
who speaks light from the darkness,  
through and for whom all things were created.

## ACKNOWLEDGEMENT

To Mom, Dad, and Grace, I would not be here without you! You have always supported my wellness and education in so many ways. Thank you for always reminding me about what is truly important in life.

To Melody, thank you for being the best partner and friend. Life is short, and there is no one I would rather spend it with. I am excited to experience many more beautiful things with you.

To my previous teachers Steve Everett, Keith Gullledge, and Carl Henriksen, thank you for instilling in me a love for systems, physics, and mathematical modeling. With every day, I realize more that my research is simply a reflection of your passion for teaching.

To all of my labmates, thank you for being the best lab that I could have. From the *many* hours of Super Smash Bros Ultimate and Mario Kart 8, to the day to day shennanigans, you have made my time at UPenn truly valuable.

To John and Harang, thank you guys for being the best. You are good, kind, and amazing friends that inspire me to better myself. Thank you, and stay in touch!

And finally, to Dani. Thank you for allowing me to pursue my crazy ideas to the fullest. It is no exaggeration to say that this thesis would not have been possible without you, your encouragement, and your support. But what the words and images in this document fail to show are the countless interactions in which you have taught me to be a better citizen of science. Thank you for being an example and an inspiration.

# ABSTRACT

## COMPLEX SYSTEMS ENGINEERING: DESIGNING ADVANCED FUNCTIONS IN DYNAMICAL AND MECHANICAL SYSTEMS

Jinsu Kim

Dani S. Bassett

From computation in neural networks to allostery in proteins, numerous natural and artificial systems are comprised of many interacting parts that give rise to advanced functions. To study such complex systems, a diverse array of interdisciplinary tools have been developed that relate the interactions of existing systems to their functions. However, engineering the interactions to perform designed functions in novel systems remains a significant challenge due to the nonlinearities in the interactions and the vast dimensionality of the design space. Here we develop design principles for complex dynamical and mechanical systems at the lowest level of their microstate interactions. In dynamical neural systems, we use methods from control theory and dynamical systems theory to mathematically map precise patterns of neural connectivity to the control of neural states in human and non-human brains (Chapter 2) and to the learning of computations on internal representations in artificial recurrent neural networks (Chapter 4). In mechanical systems, we use methods from algebraic geometry and dynamical systems to mathematically map precise patterns of mechanical constraints to design shape changes as a minimal model of protein allostery and cooperativity (Chapter 6) and to engineer mechanical metamaterials that possess arbitrarily complex shape changes (Chapter 8). These intuitive maps allow us to navigate previously unexplored design spaces in nonlinear and high-dimensional regimes, enabling us to reverse engineer form from function in novel complex systems that have yet to exist.

# TABLE OF CONTENTS

ACKNOWLEDGEMENT . . . . .	iv
ABSTRACT . . . . .	v
LIST OF TABLES . . . . .	xiii
LIST OF ILLUSTRATIONS . . . . .	xvii
PREFACE . . . . .	xviii
CHAPTER 1 : Introduction . . . . .	1
1.1 More is Different: Complexity in Many-Body Systems . . . . .	1
1.2 The Many Forms and Languages of Complexity . . . . .	2
1.3 Modeling the Microstates of Existing Systems . . . . .	3
1.4 Why Complex Systems Design is Difficult . . . . .	4
1.4.1 Nonlinearity Impedes Prediction and Design . . . . .	4
1.4.2 The Design Space is a Really Large and Complex Network . . . . .	6
1.5 The Emerging Paradigm of Complex Systems Engineering . . . . .	8
CHAPTER 2 : Role of Graph Architecture in Controlling Dynamical Networks . .	11
2.1 Motivation . . . . .	11
2.2 Mathematical Framework . . . . .	12
2.3 Predicting Control Energy . . . . .	15
2.4 Determinant of the Driver-to-Non-Driver Network . . . . .	17
2.5 Identifying Energetically Favorable Control Nodes . . . . .	18
2.5.1 Topological Contributors to Control Energy. . . . .	19
2.5.2 Energetically Favorable Driver-Non-Driver sets. . . . .	20
2.6 Complex Brain Networks are Energetically Favorable . . . . .	20

2.7	Network Manipulation to Facilitate Control . . . . .	22
2.8	Contribution and Future Directions . . . . .	23
2.9	Conclusion . . . . .	26
CHAPTER 3 : Appendix to Role of Graph Architecture in Controlling Dynamical		
	Networks . . . . .	27
3.1	Connectome Data . . . . .	27
3.2	Selection of Energetically Most & Least Favorable Non-Drivers . . . . .	29
3.3	First-Order Energy Approximation . . . . .	30
3.4	Mathematical Framework: Second-Order Energy Approximation . . . . .	43
3.5	Validity of the First-Order Approximation . . . . .	48
3.6	Comparison of Results for Directed & Undirected Networks . . . . .	48
3.7	Differences in Network Density do not Drive Differences in Results Between Networks . . . . .	51
3.8	Dealing with Driver & Non-Driver Allocations that Yield Unreachable States	55
3.9	Matrix Scaling Retains System Properties . . . . .	56
3.10	Retaining Goodness of Approximation for Scaled Matrices . . . . .	57
3.11	Analysis & Data Acquisition Retains Significant Biological Consistency Be- tween Networks . . . . .	59
3.12	Value of Mathematically Formalizing the Intuitive Concept of Differential Connectivity . . . . .	61
3.13	Validity of the Second-Order Approximation . . . . .	62
3.14	Determinant of the Second-Order Connectivity Matrix Scales the Control Energy . . . . .	63
3.15	Most & Least Energetically Favorable Driver-Non-Driver Sets in Brain Con- nectomes Using Second-Order Approximation . . . . .	65
3.16	Brain Networks of Increasingly Complex Species Have More Energetically Favorable Second-Order Organization of Connectivity Features . . . . .	66

3.17 Network Manipulation to Facilitate Control Using the Second-Order Approximation . . . . .	66
3.18 Maximizing the Determinant Through Node Selection . . . . .	67
3.19 Maximizing the Determinant Minimizes Average Control Energy for $A_{21}$ with Fixed Frobenius Norm . . . . .	68
CHAPTER 4 : Teaching Recurrent Neural Networks to Infer Global Structure . . .	71
4.1 Motivation . . . . .	71
4.2 Mathematical Framework . . . . .	73
4.3 Learning a Translation Operation by Example . . . . .	75
4.4 Learning to Infer Bifurcation Structure by Example . . . . .	77
4.5 Mechanism of how Operations are Learned . . . . .	79
4.6 Bifurcation Normal Forms & Non-Dynamical Time Series . . . . .	82
4.7 Simultaneous Learning of Multiple Operations . . . . .	84
4.8 Discussion . . . . .	85
CHAPTER 5 : Appendix to Teaching Recurrent Neural Networks to Infer Global Structure . . . . .	87
5.1 Reservoir Dynamics . . . . .	87
5.2 Form of Dynamical Equations . . . . .	89
5.3 Evaluation of the Jacobian . . . . .	92
5.4 Bias Term & Equilibrium Point Selection . . . . .	93
5.5 Tracking the Reservoir Fixed Points . . . . .	93
5.6 Poincaré Sections & the Period Doubling Bifurcation Diagram . . . . .	95
5.7 Simulation Parameters . . . . .	96
5.8 Simulation Method . . . . .	98
5.9 Training for tanh . . . . .	100
5.10 Training for Wilson-Cowan . . . . .	101
5.11 Truncation of the Block-Hessenberg Matrix . . . . .	102



5.12	Simulation of the Jansen Linkage . . . . .	105
5.13	Quantifying Prediction Accuracy of Lorenz Computations . . . . .	106
5.14	Translations, Transformations, & Bifurcations with Wilson-Cowan Networks	108
5.15	Translation in Multiple Directions . . . . .	109
5.16	Different Types of Transformations . . . . .	110
CHAPTER 6 : Conformational Control of Mechanical Networks . . . . .		112
6.1	Motivation . . . . .	112
6.2	Network Connectivity & Mathematical Framework . . . . .	113
6.3	Conic Sections & Overlaps of Bipartite Networks . . . . .	115
6.4	Network Design Through Judicious Constraint Placement . . . . .	118
6.5	Multi-Mode Construction & States of Self-Stress . . . . .	119
6.6	Combining Network Modes for Potential Applications . . . . .	121
6.7	Design of Large Displacements & Bistable Networks . . . . .	122
6.8	Discussion . . . . .	125
CHAPTER 7 : Appendix to Conformational Control of Mechanical Networks . . .		127
7.1	Defining a Projection from Solution Coordinates to Spatial Coordinates . .	127
7.1.1	Case 1: Number of Solution Coordinates: $m = d$ . . . . .	127
7.1.2	Case 2: Number of Solution Coordinates: $m = d - 1$ . . . . .	128
7.2	Characterizing the Set of All Solution Spaces . . . . .	129
7.3	Expanding on the Judicious Constraint Process . . . . .	132
7.4	Rigidity Matrix Dimensions, Spaces, & Degrees of Freedom . . . . .	133
7.5	Implications of States of Self-Stress in Infinitesimal & Finite Motions . . . .	134
7.5.1	Self-Stress in the Infinitesimal Regime . . . . .	135
7.5.2	Self-Stress in the Finite Regime . . . . .	135
7.6	Designing & Analyzing Finite Motions in Networks with Self-Stress . . . . .	138
7.7	Combining Networks with Repeating Modules . . . . .	140
7.7.1	Applying the Higher-Order Derivative Test for Finite Motion . . . . .	141

7.7.2	Combining Networks Through Judicious Constraint Placement . . .	143
7.8	Avoiding States of Self Stress in 3 Dimensions . . . . .	145
7.9	Consideration of Non-Bipartite Edges in Modules . . . . .	147
7.10	Network Combinations for Finite Motions . . . . .	150
7.11	Tristable Networks Using Intersections of Finite Motion Solution Spaces . .	151
CHAPTER 8 : Nonlinear Dynamics & Chaos in Conformational Changes of Me-		
	chanical Metamaterials . . . . .	152
8.1	Motivation . . . . .	152
8.2	Mathematical Framework . . . . .	154
8.2.1	Constraint Counting . . . . .	154
8.2.2	Defining the Set of Motions . . . . .	155
8.2.3	Constraint Counting Revisited . . . . .	156
8.2.4	Instantiating & Simulating Networks . . . . .	157
8.2.5	Motivating Statement & Outline . . . . .	159
8.3	A 4-Bar Linkage Example . . . . .	160
8.3.1	Combining Units Acts as a Map Iteration . . . . .	160
8.3.2	Visualizing Map Iteration as a Cobweb Plot . . . . .	162
8.4	Network Conformation is Known at Fixed Points . . . . .	164
8.5	Folding Sequence is Determined by Stability . . . . .	164
8.6	Designing Network Shape . . . . .	166
8.6.1	Motivating the Unit Design Procedure . . . . .	167
8.6.2	The Unit Design Procedure . . . . .	168
8.6.3	Combining Designed Units With Map Iteration . . . . .	172
8.6.4	Motivating the Network Design Procedure . . . . .	173
8.6.5	Selecting Units That Yield Global Network Shape . . . . .	174
8.7	Designing the Conformational Sequence Using Stability . . . . .	176
8.7.1	Motivating Stability Design . . . . .	177
8.7.2	Searching the Parameter Space . . . . .	177

8.8	Superstability & the Mechanical AND Gate . . . . .	179
8.8.1	Motivating Superstable Convergence . . . . .	179
8.8.2	Utilizing Superstable Convergence . . . . .	179
8.9	Period Doubling Route to Mechanical Chaos . . . . .	181
8.9.1	Motivating Chaotic Divergence . . . . .	181
8.9.2	Designing Chaotic Divergence . . . . .	182
8.10	Period Three Implies Mechanical Chaos . . . . .	183
8.10.1	Motivating 3-cycle Units . . . . .	183
8.10.2	A 3-cycle Unit and Sharkovsii's Theorem . . . . .	183
8.11	Constructing Physical Networks . . . . .	185
8.12	Elasticity & Signal Propagation in the Mechanical AND gate . . . . .	186
8.13	Discussion . . . . .	188

## CHAPTER 9 : Appendix to Nonlinear Dynamics & Chaos in Conformational Changes

	of Mechanical Metamaterials . . . . .	190
9.1	Single Module Design: Infinitesimal Motion . . . . .	190
9.2	Single Module Design: Finite Displacement . . . . .	191
9.3	Analytic Form of the Iterated Map . . . . .	192
9.4	Condition for a Conformational Motion to Act as a Map . . . . .	194
9.5	Numerically Characterizing Chaos: Lyapunov Exponent . . . . .	195
9.6	Maps of Physical Linkage & 3D-Printed Modules . . . . .	196
9.7	Construction & Map of Origami Module . . . . .	197
9.8	Edge Lengths are Determined by Added Node Placement . . . . .	198
9.9	Combining Units Only Merges Nodes Between Units . . . . .	199
9.10	Edge Lengths of All Network Examples . . . . .	200
9.11	Elasticity in the Superstable Mechanical AND Gate . . . . .	204
9.12	Elaboration of Unit Cell Topology . . . . .	206
9.13	Chaos Remains Generic to Small Changes in Bond Length . . . . .	207

CHAPTER 10 : Concluding Remarks . . . . .	209
10.1 Summary of Our Process for Extracting Design Principles . . . . .	209
10.2 Key Concepts for Complex Systems Engineering . . . . .	210
10.2.1 Preservation of the Microstate Interactions . . . . .	210
10.2.2 Nullspaces of Design are Often Immune to Complexity . . . . .	211
10.2.3 A New Twist on an Old Classic: Complex Systems Engineering Wel-	
comes You! . . . . .	212
BIBLIOGRAPHY . . . . .	212

## LIST OF TABLES

TABLE 1 :	Simulation parameters . . . . .	98
-----------	---------------------------------	----

## LIST OF ILLUSTRATIONS

FIGURE 1 :	Network Control of the Drosophila, Mouse, & Human Connectomes.	13
FIGURE 2 :	The Simplified Network Reasonably Predicts Control The Energy.	15
FIGURE 3 :	Geometric Intuition & Control Energies of First-Order Networks.	17
FIGURE 4 :	Topological Characteristics & Energetic Performance of Networks.	18
FIGURE 5 :	Energetically Favorable Organization of Network Topology. . . . .	21
FIGURE 6 :	Reducing Minimum Control Energy Through Edge Deletion. . . . .	22
FIGURE 7 :	Similar Accuracy of First-Order Energy Approximation. . . . .	49
FIGURE 8 :	Similar Distributions of Control Energy & Topology. . . . .	50
FIGURE 9 :	Similar Energetically Favorable Organization. . . . .	50
FIGURE 10 :	The Order of the Correlation is Maintained between Species. . . . .	51
FIGURE 11 :	Similar Changes in Control Energy for Edge Deletions. . . . .	52
FIGURE 12 :	Percent Error in Energy Improves for Increasing Densities. . . . .	52
FIGURE 13 :	Non-Driver Selection Based on the Topology-Dependent Term Yields Significantly Different Energetic Performance across Densities. . . . .	53
FIGURE 14 :	Non-Monotonic Relationship between Spearman Rank Correlation Coefficient & Network Density. . . . .	54
FIGURE 15 :	Similar Distributions for Changes in Control Energy for Energetically Favorable Edge Deletions across Varying Densities. . . . .	54
FIGURE 16 :	Higher Number of Human Connectomes Share Energetically Favorable Edges for Deletion than Randomly Selected Edges. . . . .	60
FIGURE 17 :	Differential Connectivity Difficult to Identify with Increasing Net- work Size. . . . .	61
FIGURE 18 :	The Second-Order Energy Approximation Offers a Reasonable Pre- diction for the Full Network's Control Energy for Higher Non- Driver Fractions. . . . .	63

FIGURE 19 : Geometric Example of Simplified, Second-Order Networks with Corresponding Control Energies. . . . .	64
FIGURE 20 : Topological Characteristics & Energetic Performance of Networks using the Second-Order Approximation. . . . .	65
FIGURE 21 : Energetically Favorable Organization of Second-Order Topological Features in Networks. . . . .	66
FIGURE 22 : Modifying the Connectomes to Decrease Minimum Energy Using both Driver to Non-Driver, as well as Non-Driver to Non-Driver Connections. . . . .	67
FIGURE 23 : Decreasing Average Energy as a Function of Increasing Determi- nant in Brain Networks. . . . .	68
FIGURE 24 : Representing Chaotic Attractors with Reservoirs. . . . .	73
FIGURE 25 : Learning & Extrapolating Translations & Transformations by Ex- ample. . . . .	76
FIGURE 26 : Inferring & Extrapolating the Bifurcation of the Lorenz. . . . .	78
FIGURE 27 : Changing the Control Parameter Changes the Reservoir Dynamics to Manipulate Representations. . . . .	80
FIGURE 28 : Inferring Bifurcation Normal Forms & Extrapolating Kinematic Trajectories. . . . .	82
FIGURE 29 : Flight of the Lorenz. . . . .	84
FIGURE 30 : Predicted Change in Reservoir States Given a Change in Control Parameter. . . . .	103
FIGURE 31 : Tanh & Wilson-Cowan Attractor Similarity. . . . .	107
FIGURE 32 : Translation of the Lorenz Representation Using Wilson-Cowan Os- cillator Networks. . . . .	108
FIGURE 33 : Transformation of the Lorenz Representation Using Wilson-Cowan Oscillator Networks. . . . .	108

FIGURE 34 : Bifurcation of the Lorenz Representation Using Wilson-Cowan Oscillator Networks. . . . .	109
FIGURE 35 : Translation of the Lorenz Representation in all Three Spatial Directions. . . . .	110
FIGURE 36 : Transformation of the Lorenz Representation Using Stretch & Shear in Several Spatial Directions. . . . .	111
FIGURE 37 : Graphical Representations of Maxwell Frames. . . . .	113
FIGURE 38 : Solution Space of Unspecified Nodes is Determined by the Specified Nodes. . . . .	116
FIGURE 39 : Construction & Control of Frames with Specified Outward Motion. . . . .	118
FIGURE 40 : Intersections of Solution Spaces for Multiple Non-Rigid Motions. . . . .	120
FIGURE 41 : Combining Network Motions by Merging Nodes & Adding Edges. . . . .	121
FIGURE 42 : Designing Finite Motions & Bistable Networks with Cooperativity. . . . .	123
FIGURE 43 : Solution Space Intersections. . . . .	132
FIGURE 44 : Constructing a Network that is Pre-Stress Stable. . . . .	138
FIGURE 45 : Constructing a Network at the Intersection of Branching. . . . .	139
FIGURE 46 : Combination of Identical Modules with Nonlinear Symmetries Through Node Merging. . . . .	140
FIGURE 47 : Construction of Large Network Motions Through the Judicious Coupling Between Non-Intersecting Modules. . . . .	144
FIGURE 48 : Non-Planar Judicious Constraint of 5 Nodes in $d = 3$ . . . . .	146
FIGURE 49 : Constraining Networks with Non-Bipartite Edges. . . . .	149
FIGURE 50 : Combining Networks with Finite Positions. . . . .	150
FIGURE 51 : Tristable Networks Using Solution Space Intersections. . . . .	151
FIGURE 52 : Constraints & Conformational Motions. . . . .	154
FIGURE 53 : Motivation for the Results. . . . .	159
FIGURE 54 : Conformational motion as a map. . . . .	160



FIGURE 55 : Combine Units by Merging Nodes. . . . .	161
FIGURE 56 : Shape & Folding Sequence of Iterated Maps. . . . .	162
FIGURE 57 : Properties of Unit Design. . . . .	166
FIGURE 58 : Designing Unit Geometry at Fixed Points. . . . .	168
FIGURE 59 : Representing the Combining of Designed Units as an Iterated Map.	170
FIGURE 60 : Designing Precise Network Geometry. . . . .	174
FIGURE 61 : Designing the Sequence of Conformational Change. . . . .	176
FIGURE 62 : Superstability & Extreme Localization Through Faster-Than-Exponential Convergence. . . . .	179
FIGURE 63 : Mechanical Chaos. . . . .	181
FIGURE 64 : A 3-Cycle Unit. . . . .	184
FIGURE 65 : Physical Construction of Networks. . . . .	185
FIGURE 66 : Elastic Deformations in Superstable Networks. . . . .	187
FIGURE 67 : Decomposition of a Conformational Motion Into Valid Map Segments.	194
FIGURE 68 : Trajectory Divergence at Chaos. . . . .	195
FIGURE 69 : Maps of Physical Networks. . . . .	196
FIGURE 70 : Origami Sheet Construction & Map. . . . .	197
FIGURE 71 : Edge Lengths are Determined by Added Node Placement. . . . .	198
FIGURE 72 : Combining Units Only Merges Nodes Between Units. . . . .	199
FIGURE 73 : Decomposition and Edge Lengths of the 4-Bar Linkage Example.	200
FIGURE 74 : Decomposition and Edge Lengths of the Designed Quadrifolium. .	201
FIGURE 75 : Decomposition and Edge Lengths of the Designed Superstable Net- work. . . . .	202
FIGURE 76 : Decomposition and Edge Lengths of the Designed Chaotic Network.	203
FIGURE 77 : Evolution of an Elastic Mechanical AND Gate. . . . .	204
FIGURE 78 : Physical Construction of the Mechanical AND Gate. . . . .	205
FIGURE 79 : Robustness of Chaos to Random Parameter Perturbations. . . . .	208

## PREFACE

A complex system is comprised of many interacting parts that produce incredible behaviors. From the information processing that arises from interacting neurons to the chemical reactions catalyzed by interconnected amino acids in enzymes, nature testifies to the staggering complexity that can emerge from simple interactions. In an attempt to understand this emergence, an equally staggering set of methods and conceptual paradigms have been developed that relate complex interactions to emergent behaviors in existing systems.

But what about systems that have yet to exist? How can we harness the seemingly infinite potential of many-body interactions to *engineer* our own complex systems with designed properties? The answers to these questions are hindered by a multitude of difficulties, chief among which are the large number of parts and the complexity of their interactions and organization. This dissertation is a catalog of my attempts to navigate these difficulties to uncover design principles in two principle types of systems: those that evolve in time such as neural networks (dynamical), and those that evolve in space such as proteins (mechanical).

Beyond the specific text and results of any one chapter, the main goal of this dissertation is to impress upon the reader that complexity and emergence, in addition to being topics of great scientific inquiry, can also be harnessed for the purposes of design and engineering.

## CHAPTER 1 : Introduction

### 1.1. More is Different: Complexity in Many-Body Systems

Many systems that are coupled in time, or *dynamical*, undergo a qualitative change in behavior with increasing number of components. A single neuron—albeit quite complex in its own right (Martini, 2007)—can be accurately modeled using differential equations (Hodgkin and Huxley, 1952). The nematode *Caenorhabditis elegans* with 302 neurons is more complex (White et al., 1986), enabling functions such as chemosensation (Bargmann, 2006), digestion (Avery and You, 2018), and navigation (Qin and Wheeler, 2007). The human brain comprising 80 billion neurons is yet more complex (Herculano-Houzel, 2009), enabling advanced functions such as the object representation (Tacchetti et al., 2018), accumulation of evidence (Eguíluz et al., 2015), and working memory (Courtney et al., 1997).

Additionally, many systems that are coupled in space, or *mechanical*, also undergo a qualitative change in behavior with increasing numbers of components. A single amino acid—albeit quite complex in its own right (Lehninger et al., 2005)—possesses a well-studied repertoire of chemical properties (Creighton, 1993). Chaining several of these amino acids into a polypeptide produces spatially complex secondary structures such as  $\alpha$ -helices and  $\beta$ -sheets. Bringing together multiple polypeptides forms complex quaternary structures such as hemoglobin, which saturates and transports oxygen in the bloodstream (Lukin et al., 2003; Lukin and Ho, 2004) by changing its geometry to bind oxygen.

In these and many other examples in nature, increasing the number of components gives rise to qualitatively different and advanced functions which cannot be understood as an extrapolation of a single part. Hence, understanding the laws that govern the behavior of the parts and their interactions is only the first step in engineering complex systems. In the words of P. W. Anderson, “The ability to reduce everything to simple fundamental laws does not imply the ability to start from those laws and reconstruct the universe” (Anderson, 1972). Perhaps more succinctly put in the title of this specific work: “More is *Different*”.

## 1.2. The Many Forms and Languages of Complexity

To understand precisely *how* more is different, complexity has enjoyed a long and rich history of study across many disciplines and in many different forms. At the macroscopic scale, one such form is *self-organization*: the creation of global order from local interactions. Examples include the synchronization of oscillators (Acebrón et al., 2005) and the concentration of energy density in ecological systems (Odum, 1988). At the mesoscopic scale, another such form is *pattern formation*: the emergence of ordered structure from local interactions. Examples range from periodic wave patterns in biological tissue during morphogenesis (Gierer and Meinhardt, 1972) to the fractal and time-evolving structures of cellular automata (Wolfram, 1984). At the microscopic scale, complexity is everywhere from information processing in artificial neural networks (LeCun et al., 2015) to the intricate pathways of gene regulatory networks (Karlebach and Shamir, 2008).

To study such a wide expanse of systems across many fields, the language used to describe complexity is equally vast. In dynamical systems—the study of time-evolving equations—complex patterns may be called *attractors*, the self-organization onto said attractors are determined by *dynamical equations*, and qualitatively different behaviors can take the form of *bifurcations* (Strogatz, 2018). In mechanics—the study of spatially constrained equations—complex patterns are called *configuration manifolds*, the self-organization onto said manifolds are determined by *Hamiltonians*, and qualitatively different behaviors can take the form of *kinematic singularities* (Lurie, 2002). One may similarly choose to view complexity through the lens of information theory and statistical mechanics (Jaynes, 1957), graph theory (McCabe, 1976), and any number of other methods and paradigms of thought.

Through such monumental efforts, significant progress has been made in developing methods and paradigms of thought for understanding complex systems. While by no means exhaustive, we focus on the two aforementioned cases of dynamical and mechanical systems throughout this dissertation.

### 1.3. Modeling the Microstates of Existing Systems

Provided a system where all of the components and their interactions are given, there exists a plethora of tools to determine its behavior. In a dynamical system whose behavior is determined by a set of time-dependent equations, we can numerically simulate the evolution of the system forward in time using a vast array of integration techniques (Cash and Karp, 1990) that incorporate real-world factors such as time delays (Kuang, 2012) and stochasticity (Arnold, 1974). In a mechanical system whose behavior is determined by energetics and spatially-constrained equations of motion, we can run molecular dynamics simulations (Rapaport and Rapaport, 2004) and finite element analysis (Zienkiewicz et al., 1977) to observe the system’s behavior. Hence, given full knowledge of a system’s microstates and interactions, it is theoretically possible to simulate the behavior of the microstates.

Of course, these approaches are not without drawbacks, one of which is the significant difficulty of capturing and simulating sufficient microscopic detail for accurate modeling (Randi and Leifer, 2020). To mitigate this drawback, many coarse-grained models reduce the complexity of a single component. An example in neural systems is the FitzHugh-Nagumo model: a 2 variable simplification of the Hodgkin-Huxley model that produces action potentials (FitzHugh, 1961). An example in mechanical systems is the Gaussian model: a 2-coordinate simplification of an amino acid using its position and orientation (Micheletti et al., 2004). Another approach is to model the average state of populations such as the Wilson-Cowan model for neurons (Wilson and Cowan, 1972) or the rigid cluster analysis for proteins (Potestio et al., 2009). Other forms of coarse graining involve mathematical dimensionality reduction by taking advantage of the interaction topology as in external equitable partitions (Schaub et al., 2016) or more data-driven methods such as dynamic mode decomposition (Schmid, 2010).

Hence, with several caveats, a myriad of tools exist to study and simulate the microstates of existing systems. Unfortunately, the ability to simulate these fundamental laws does not imply the ability to use them for construction.

## 1.4. Why Complex Systems Design is Difficult

### 1.4.1. Nonlinearity Impedes Prediction and Design

Despite the existence of such sophisticated tools that describe the behavior of existing systems, the inverse problem of designing novel behaviors in engineered systems remains a significant challenge. At first glance, it would appear that the main barrier is the sheer magnitude of the design space: given an already large number  $n$  of components, the space of possible interactions is  $n^2$ . However, what makes the problem truly difficult is our inability to *predict* the behavior of systems at unobserved states and at unobserved patterns of interactions. Said another way, it is generally not the case that given a few candidate systems for which we know the behaviors, we can know the behavior of other systems that exist in different states or have different patterns of interactions.

As an illustration, let us consider the behavior of a 1-state dynamical system given by

$$\frac{d}{dt}x = -1 + x^2, \tag{1.1}$$

and consider three scenarios with slightly different initial states around  $x(0) = 1$ , under the naive hope that nearby initial states might yield similar behaviors.

- If we begin our system at  $x(0) = 1$ , the rate of change of  $x$  will be 0 such that the system will stay at  $x(t) = 1$ .
- If we begin our system at  $x(0) = 0.9$ , the rate of change of  $x$  will be negative, such that the system eventually decays to  $x(t) = -1$ .
- If we begin our system at  $x(0) = 1.1$ , the rate of change of  $x$  will always be positive, such that the system blows up to  $x(t) \rightarrow \infty$ !

Hence, even in this 1-state system, while it is simple to simulate its behavior at specific initial conditions, it is far more difficult to predict the outcome of untested initial conditions.

This particular form of unpredictability goes by the name of *nonlinear*. By nonlinear, all we mean is that the outputs do not scale with the inputs. In the previous 1-state dynamical system in Eq. 1.1, we found that if the initial state is  $x(0) = 1$ , then the final state at  $t \rightarrow \infty$  is also  $x^* = 1$ . If the system is *linear*, then an initial state of  $x(0) = \alpha \cdot 1$  should yield a final state of  $x(t) = \alpha \cdot x^*$ . In such an ideal scenario, the design problem is simple: if we desire a final state  $x(t) = \alpha x^*$ , then we have only to start the system at  $x(0) = \alpha \cdot 1$ . Unfortunately, very few natural systems are so ideally linear.

To circumvent this problem of nonlinearity, a wide range of tools has been developed. Some methods use special mathematical tools called *composition* or *Koopman* operators to represent a nonlinear dynamical system as a linear system with more, and often infinite, state variables (Williams et al., 2015; Budišić et al., 2012; Brunton et al., 2016). Other methods in nonlinear dynamics involve using geometrical analysis and analytical tools to study the long-term behavior of low-dimensional nonlinear systems (Strogatz, 1994). There also exist numerical tools to perform successive local approximations of the nonlinear system as a linear one within a small neighborhood of the current state (Cornelius et al., 2013b), or linearizations about a periodic trajectory (Hespanha, 2018). Still other methods take successive analytical approximations using higher-order convolutions to write down a closed-form expression of the state evolution through Volterra kernels (Brockett, 1976). The goal is the same: overcome nonlinearity to write the output as a function of the input. However, the existence of such a breadth of approaches should indicate to the reader that nonlinearity, while a simply posed problem, does not yet have a general solution.

Hence, the first challenge when designing complex systems is to tame the nonlinearities to attain predictive design power. In this dissertation, we will use whatever approaches yield the greatest designability and the most conceptually tractable intuition. Often, the reader will see methods at the heart of dynamical systems used as design principles for mechanical networks, such as in Chapter 8. Other times, the reader will come across geometric intuitions for designing dynamical networks, such as in Chapter 2.

#### 1.4.2. The Design Space is a Really Large and Complex Network

The lack of predictability due to nonlinearity poses a very real problem. This is because each component of a system—such as a neuron’s membrane potential or an amino acid’s spatial coordinates—is a real, continuous number. In a system of  $n$  components, the state space of the system is of order  $\mathbb{R}^n$ , which requires exponentially more compute time to sample densely with increasing  $n$ . Yet, the true design space is exponentially larger because such systems have  $n^2$  possible interactions. Just as the synaptic strength between neurons or the bond strength between amino acids is a continuous quantity, so too is the strength of interaction between each pair of components. Hence, the design space of interactions resides not in  $\mathbb{R}^n$ , but rather in  $\mathbb{R}^{n \times n}$ .

As a result, many exact and fully nonlinear design principles that exist at the state space level deal primarily with very small  $n$ . For many dynamical systems, we possess an analytical and geometrically intuitive understanding about how the interactions between dynamical components affect their behavior up to  $n = 3$  (Strogatz, 1994). However, this understanding is by no means complete. For example, as per Hilbert’s 16th problem, the maximum number of limit cycles that can exist for a polynomial 2-dimensional dynamical system is still unknown (Ilyashenko, 2002), and we are still discovering new 3-dimensional chaotic systems (Li and Sprott, 2018). For mechanical systems, the study of *kinematic synthesis* gives exact and analytical solutions for the behavior of mechanical linkages comprising a few rigid bars connected by rotating joints (Hartenberg and Danavit, 1964). When designing the nonlinear interactions between more components, we typically resort to numerical optimization methods (Werbos, 1990; Baskar and Bandyopadhyay, 2019; Rocks et al., 2017b).

In addition to the sheer magnitude of the interaction space, it is also full of nonlinear organization. One example is the importance of the *topology* of the interaction network. Informally, the term topology refers to the precise pattern of interactions that are present or absent, such as the structural connections in the brain (Alexander et al., 2007). Mathematically, topology refers to a broad field of study regarding the preservation of structures



such as cavities under continuous deformation, and branches of the field such as algebraic topology help to quantify higher-order dependencies in networks (Sizemore et al., 2018). Naturally occurring patterns include modular (Valencia et al., 2009), hierarchical (Zhou et al., 2006), and even fractal (Mandelbrot and Mandelbrot, 1982) organization.

Many approaches successfully quantify and characterize such complex organization. One common approach in network science is to define features of connectivity that are naturally observed, conceptually motivated, or mathematically derived, and compute their presence in networks. Some connectivity features that are thought to be important for communication between brain regions are the shortest path (Avena-Koenigsberger et al., 2018), the weighted sum of paths of various lengths (Estrada and Hatano, 2008), and small-worldness (Watts and Strogatz, 1998b). Other features focus on precise local structures such as *motifs* (Sporns et al., 2004), hubs (Hwang et al., 2017), and coordination number (Silbert, 2010). Such an approach has been fruitful for dynamical networks such as neural systems (Bassett and Sporns, 2017) and mechanical systems such as packings of particles (Richard et al., 2020).

Hence, far from remaining restrained by the difficulties that arise from nonlinearity and dimensionality, our growing fascination with the complex has given rise to an incredibly diverse set of approaches. The aforementioned list of approaches is by no means exhaustive, but is rather intended to serve as a demonstration of the potential that exists in complexity to warrant such extensive study. In the words of Wilson J. Rugh (Rugh, 1981):

“When confronted with a nonlinear systems engineering problem, the first approach usually is to linearize; in other words, to try to avoid the nonlinear aspects of the problem. It is indeed a happy circumstance when a solution can be obtained in this way. When it cannot, the tendency is to try to avoid the situation altogether, presumably in the hope that the problem will go away. Those engineers who forge ahead are often viewed as foolish, or worse. Nonlinear systems engineering is regarded not just as a difficult and confusing endeavor; it is widely viewed as dangerous to those who think about it for too long.”

## 1.5. The Emerging Paradigm of Complex Systems Engineering

In spite of the associated danger, the need to engineer complex systems has been too great to ignore. One such need is in neural engineering for therapies to correct dysfunctions and disorders such as drug resistant epilepsy (Kwan et al., 2010), which often requires alternative surgical interventions such as resection and electrical stimulation (Duncan et al., 2006). Recent work has assisted in the identification of seizures (Bernabei et al., 2021), characterized network properties throughout a seizure (Scheid et al., 2021), and even predicted the surgical outcome of patients using simulated virtual resections (Kini et al., 2019). In tandem, recent work has also applied methods from control theory to model the effect of stimulation in neural systems (Stiso et al., 2019; Betzel et al., 2016b; Gu et al., 2017). Can we bridge these two lines of work to determine an intuitive, first-principles theory for how the complex patterns of neural connectivity determine the response to stimulation? In this dissertation, Chapter 2—adapted from (Kim et al., 2018; Kim and Bassett, 2020)—is dedicated to developing precisely this theory, where we can not only obtain a simple geometric understanding of how specific patterns of connectivity determine stimulation response, but easily use that understanding to virtually resect edges that improve the response.

Beyond biomedical and therapeutic need, the potential of engineering complex dynamical systems has led to the development of a wide range of uses, among which deep learning is a prominent example. Starting with early models of artificial neurons (McCulloch and Pitts, 1943), the field of deep learning has evolved to develop advanced methods for training large networks (Rumelhart et al., 1986; Werbos, 1990; Sutton and Barto, 2018) to perform highly sophisticated functions such as image generation (Gregor et al., 2015), language modeling (Mikolov et al., 2010), and time series prediction (Sussillo and Abbott, 2009). What is special about these examples is that they use *recurrent* neural networks (RNNs) which possess an internal representation through internal states that evolve in time as a dynamical system. This internal representation is precisely what enables the incredible computational capability of RNNs (Schäfer and Zimmermann, 2006) such as the associative memory of

Hopfield networks (Hopfield, 1982), and precisely what makes the formal analysis and design of their architecture challenging. To understand how RNNs form internal representations, concepts in dynamical systems such as *generalized synchronization* (Rulkov et al., 1995a), *invertible generalized synchronization* (Lu and Bassett, 2020), and Lyapunov exponents (Balcerzak et al., 2018) provide a mathematical understanding of memory formation and representation in RNNs. But can we further develop this understanding to map the precise network of neural connections to the formation and manipulation of these representations? The development of such a map is the focus of Chapter 4—adapted from (Kim et al., 2021)—where we train RNNs to manipulate their own internal representations, and precisely map the neural connectivity to these computations.

In parallel with complex dynamical systems is the development of design principles for complex mechanical systems. The need for precise geometric shape change is seen prominently in biological processes such as the substrate selectivity and cooperativity of proteins (Lukin and Ho, 2004) and the long-range allosteric excitation and inhibition of enzymes (Lisi and Loria, 2017). To design such geometric shape changes, many algorithmic approaches perform numerical optimizations on the network topology and interaction strength to obtain a desired response (Rocks et al., 2017b; Leman et al., 2020). But can we develop simple design principles for engineering general shape changes? The development of such principles is the main focus of Chapter 6—adapted from (Kim et al., 2019b)—where we define a simple design framework for precise changes in network geometry, and use it to provide a minimum model of protein allostery and cooperativity.

Beyond biomedical need, the potential of engineering complex mechanical system has led to the development of a fundamentally novel class of materials called *metamaterials*. These materials possess incredibly sophisticated responses ranging from extreme tunable stiffness (Hwang and Bartlett, 2018) to the algorithmic generation of complex curvature (Choi et al., 2019) with applications that range from designing locking mechanisms (Surjadi et al., 2019b) to solar sails (Fu et al., 2016b). Such materials are often constructed by combining small

units into a large material where the material’s response is designed using the geometry of the units and their coupling. To construct such a complex array of material responses, it seems almost necessary to rely on numerical optimization or to study systematic variations of a single unit (Jin et al., 2020) and measure the material’s response. But what if there were a simple theory for coupling units that somehow gave rise to a rich and nonlinear class of easily designable material responses? The development of such a theory is precisely the focus of Chapter 8—adapted from (Kim et al., 2019a)—where we demonstrate a mathematical equivalence between metamaterial design and dynamical systems. By writing the addition of units as a dynamical map iteration, we bring the full brunt of dynamical systems theory to develop a complete set of design principles for precisely engineering the geometry and folding sequence of a network, to the point of designing units that can change shape chaotically and even possess an infinite number of periodic configurations.

This dissertation is organized in two main parts. Chapters 2 and 4 focus on biologically and artificially inspired design principles in dynamical neural networks, respectively, and Chapters 6 and 8 focus on biologically and artificially inspired design principles in mechanical networks, respectively.

Through these chapters, we wish to communicate that complexity—while often convenient to simplify, coarse grain, numerically optimize, and average over—is worthy of a closer look.

## CHAPTER 2 : Role of Graph Architecture in Controlling Dynamical Networks

### 2.1. Motivation

Network systems are composed of interconnected units that interact with each other on diverse temporal and spatial scales (Newman, 2010). The exact patterns of interconnections between these units can take on many different forms that dictate how the system functions (Newman, 2003). Indeed, specific features of network topology – such as small-worldness (Watts and Strogatz, 1998a) and modularity (Simon, 1962) – can improve efficiency and robustness. Yet, exact mechanisms driving the relationship between structure and function remain elusive, hampering the analysis, modification, and control of interconnected complex systems. The relationship between interconnection architecture and dynamics is particularly important in biological systems such as the brain (Bassett and Sporns, 2016), where it is thought to support optimal information processing at cellular (Bettencourt et al., 2007) and regional (Bassett and Bullmore, 2016; Sporns and Betzel, 2016) levels. Understanding structure-function relationships in this system could inform personalized therapeutics (Barabasi et al., 2011) including more targeted treatments for drug-resistant epilepsy to make the epileptic state energetically unfavorable to maintain (Ching et al., 2012; Khambhati et al., 2016), especially due to the development of multi-site stimulation tools (Gonen et al., 2017; Mohanty and Lakshminarayanan, 2015) that allow for exponentially increasing stimulation configurations.

Existing paradigms seeking to explain how a complex network topology drives observable dynamics have advantages and disadvantages. Efforts in nonlinear dynamics define basins of attraction and perturbations driving a system between basins (Sprott and Xiong, 2015; Cornelius et al., 2013a). Efforts in network science define graph metrics and report statistical correlations with observed functions such as attention (Shine et al., 2016) and learning (Mantzaris et al., 2013; Bassett et al., 2014). Neither approach offers comprehensive analytical solutions explaining mechanisms of control. A promising paradigm that meets these challenges is linear network control theory (Kalman, 1963; Lin, 1974), which assumes that

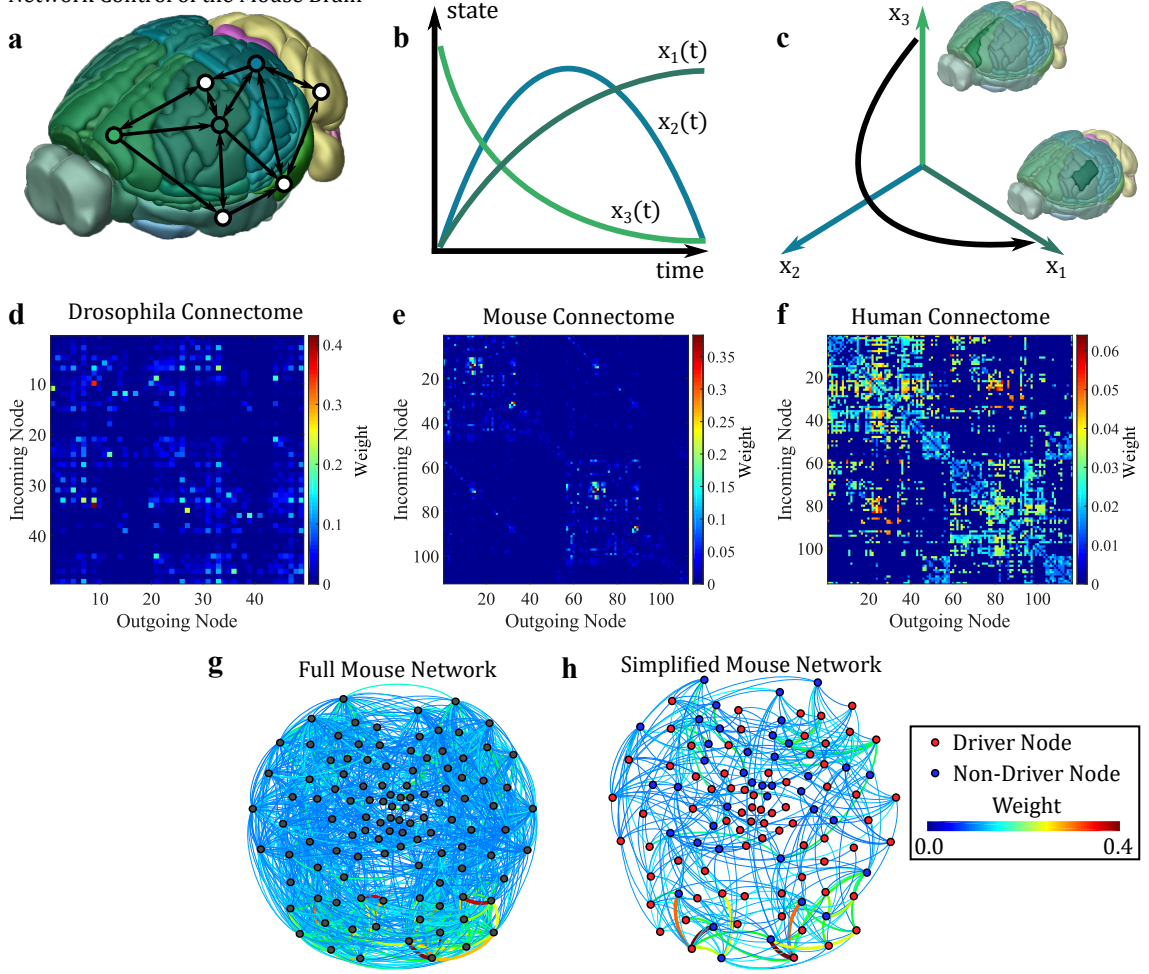
the state of a system at a given time is a function of the previous state, the structural network linking system units, and injected control energy. From this paradigm, one can identify (i) driver nodes (Liu et al., 2011; Ruths and Ruths, 2014) capable of influencing the system along diverse trajectories, and (ii) optimal inputs that move the system from one state to another with minimal cost. This latter formulation has proven useful in understanding the human brain where control points enable diverse cognitive strategies (Gu et al., 2015, 2017), facilitate efficient intrinsic activation (Betzel et al., 2016a), and inform optimal targets for brain stimulation (Muldoon et al., 2016).

While practical tools exist, basic intuitions about the network properties that enhance control have remained elusive. Here, we address this challenge by formulating a linear control problem on the bipartite subgraph linking driver nodes to non-driver nodes, which provides excellent estimates of the control of the full network. Our results include analytical derivations of expressions relating a network’s minimum control energy to its connectivity, an intuitive geometric representation to visualize this relationship, and rules for modifying edges to alter control energy in a predictable manner. While our mathematical contributions are applicable to any complex network system whose dynamics can be approximated by a linear model, we illustrate their utility in the context of networks estimated from the mouse (Oh et al., 2014; Rubinov et al., 2015), *Drosophila* (Shih et al., 2015), and human brain (Fig. 1d–f). Our results offer fundamental insights into the patterns of connections between brain regions that directly impact their minimum control energy, providing a link between the structure and function of neural systems and informing potential clinical interventions. An extension of this framework to non-bipartite graphs with corresponding results can be found in the appendix in Chapter 3.

## 2.2. Mathematical Framework

We consider a network represented by the directed graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ , where  $\mathcal{V} = \{1, \dots, n\}$  and  $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$  are the sets of network vertices and edges, respectively. Let  $a_{ij} \in \mathbb{R}$  be

# Network Control of the Mouse Brain



**Figure 1: Network Control of the Drosophila, Mouse, & Human Connectomes.** (a) A representation of the mouse brain via the Allen Mouse Brain Atlas, with a superimposed simplified network. Each brain region is represented as a vertex, and the connections between regions are represented as directed edges. (b) Example trajectories of state over time for three brain regions, where the state represents the level of activity in each region. (c) A state-space representation of activity on the mouse connectome over time, where each point on the black line represents the brain state at a point in time. (d) Connectomes represented as  $n \times n$  adjacency matrices where each  $i, j$ th element of the adjacency matrix represents the strength of the connection from node  $j$  to node  $i$  for Drosophila, (e) mouse, and (f) human. (g) The mouse connectome represented as a graph with vertices as brain regions, and edges colored by their weight, or the magnitude of the relevant element of the adjacency matrix. (h) Simplified graph representation: a bipartite subgraph containing edges linking driver vertices (red) to non-driver vertices (blue).

the weight associated with the edge  $(i, j) \in \mathcal{E}$ , and let  $A = [a_{ij}]$  be the weighted adjacency matrix of  $\mathcal{G}$ . We associate a real value (*state*) with each node, collect the nodes' states

into a vector (*network state*), and define the map  $\mathbf{x} : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^n$  to describe the evolution (*dynamics*) of the network state over time (Fig. 1a–c). We assume that a subset of  $N$  nodes, called drivers, is independently manipulated by external controls and, without loss of generality, we reorder the network nodes such that the  $N$  drivers come first. Thus, the network dynamics read as

$$\begin{bmatrix} \dot{\mathbf{x}}_{\text{d}} \\ \dot{\mathbf{x}}_{\text{nd}} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{\text{d}} \\ \mathbf{x}_{\text{nd}} \end{bmatrix} + \begin{bmatrix} I_N \\ 0 \end{bmatrix} \mathbf{u}, \quad (2.1)$$

where  $\mathbf{x}_{\text{d}}$  and  $\mathbf{x}_{\text{nd}}$  are the state vectors of the driver and non-driver nodes,  $A_{11} \in \mathbb{R}^{N \times N}$ ,  $M = n - N$ ,  $A_{12} \in \mathbb{R}^{N \times M}$ ,  $A_{21} \in \mathbb{R}^{M \times N}$ ,  $A_{22} \in \mathbb{R}^{M \times M}$ ,  $I_N$  is the  $N$ -dimensional identity matrix, and  $\mathbf{u} : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^N$  is the control input.

We will use the word *controllable* to refer to networks that are *point-to-point controllable* at time  $T \in \mathbb{R}_{\geq 0}$  if, for any pair of states  $\mathbf{x}_{\text{d}}^*$  and  $\mathbf{x}_{\text{nd}}^*$ , there exists a control input  $u$  for the dynamics Eq. (2.1) such that  $\mathbf{x}_{\text{d}}(T) = \mathbf{x}_{\text{d}}^*$  and  $\mathbf{x}_{\text{nd}}(T) = \mathbf{x}_{\text{nd}}^*$ . For a detailed discussion and rigorous conditions for the controllability of a system with linear dynamics, see (Kailath, 1980). We define the energy of  $\mathbf{u}$  as

$$E(\mathbf{u}) = \sum_{i=1}^N \underbrace{\int_0^T u_i(t)^2 dt}_{E_i},$$

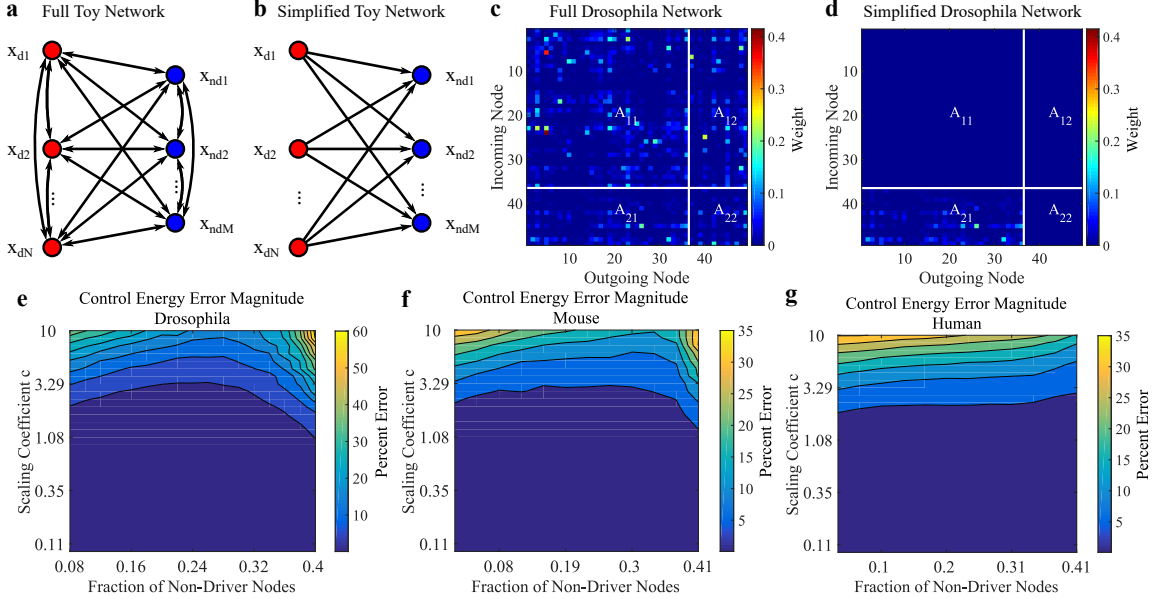
where  $u_i$  is the  $i$ -th component of  $\mathbf{u}$ . The energy of  $u_i$  can be thought of as a quadratic cost that penalizes large control inputs.

In the context of the brain, we approximate the interactions between brain regions as linear, time invariant dynamics, where a stronger structural connection between two regions represents a stronger dynamic interaction (for empirical motivation, see (Gu et al., 2015; Fernandez, 2008; Honey et al., 2009)). We specifically study the empirical inter-areal meso-scale connectomes of the mouse (112 brain regions, example schematic in Fig. 1g,h) from the Allen Brain Institute, the Drosophila (49 brain regions) (Shih et al., 2015), and a set of



human connectomes (116 brain regions) interconnected by white matter tracts (for empirical details regarding connectivity estimates, see appendix in Chapter 3).

### 2.3. Predicting Control Energy



**Figure 2: The Simplified Network Reasonably Predicts The Control Energy.** (a) Graphical representation of a *non-simplified* network of  $N$  drivers (red) and  $M$  non-drivers (blue), with directed connections between all nodes present. (b) Graphical representation of a *simplified* first-order network only containing first-order connections from drivers  $\rightarrow$  non-drivers. (c) As an example, we show the adjacency matrix for the Drosophila connectome segmented into driver  $\rightarrow$  driver  $A_{11}$ , driver  $\rightarrow$  non-driver  $A_{12}$ , non-driver  $\rightarrow$  driver  $A_{21}$ , and non-driver  $\rightarrow$  non-driver  $A_{22}$  sections for a non-simplified network as per Eq. (2.1), with randomly designated driver and non-driver nodes, and (d) the corresponding simplified network as per Eq. (2.2). (e) Percent error contour plots of the total control energy for simplified *versus* non-simplified networks as a function of the fraction of non-driver nodes and matrix scale given by  $c = \|\lambda_{\max}\|$ . For each combination of parameters, the median error magnitude to drive the networks from initial states  $\mathbf{x}_d = 0$ ,  $\mathbf{x}_{nd} = 0$  to 1000 random final states  $\mathbf{x}_{nd}^* \in (-1, 1)^M$ ,  $\mathbf{x}_d^* \in (-1, 1)^N$  along 1000 corresponding random selections of non-drivers is shown. Each contour represents a 5% interval for the (e) Drosophila, (f) mouse, and (g) human connectome.

We seek an accurate, tractable relationship between the energy required to drive a network to a specific state and its connectivity. We begin with the original, *non-simplified network* (Fig. 2a) involving edges between all nodes, and consider dynamics along the *simplified net-*

*work* (Fig. 2b) involving only edges from the driver to the non-driver nodes (for a conceptual schematic of the full and simplified *Drosophila* connectome, see Fig. 2c–d). We then derive an approximation of the minimum control energy (Lemma X.2 - X.4) by assuming that  $\mathbf{x}_d(0) = 0$ ,  $\mathbf{x}_{nd}(0) = 0$  (Assumption 1), and  $A_{11} = 0$ ,  $A_{12} = 0$ , and  $A_{22} = 0$  (Assumption 2) in Eq. (2.1), which reads as

$$E(\mathbf{u}) = 12(\mathbf{x}_{nd}^* - \frac{1}{2}A_{21}\mathbf{x}_d^*)^T(A_{21}A_{21}^T)^{-1}(\mathbf{x}_{nd}^* - \frac{1}{2}A_{21}\mathbf{x}_d^*) + \mathbf{x}_d^{*T}\mathbf{x}_d^*. \quad (2.2)$$

We make Assumption 1 because we are interested in the *change* in brain state through control, and consider initial conditions  $\mathbf{x}_d(0) = 0$ ,  $\mathbf{x}_{nd}(0) = 0$  to be a neutral baseline. Because Eq. (2.2) only involves edges from driver to non-driver nodes, we call Eq. (2.2) a first-order approximation to the minimum control energy of the non-simplified network Eq. (2.1). Importantly, this approximation requires at least as many driver nodes as non-driver nodes for  $A_{21}A_{21}^T$  to be invertible (i.e.  $N \geq M$ ). To assess the accuracy of our expression, we look to classic results in the mathematical theory of systems and control (Kailath, 1980), where the spectral properties of the *reachability Gramian*  $W_R(0, T) = \int_0^T e^{At}BB^Te^{A^Tt}dt$  quantify the minimum amount of energy (Section XI A 2) to control the non-simplified network Eq. (2.1).

In these brain networks, we observe that the first-order energy approximation is accurate across a range of parameters, which are the magnitude of the adjacency matrix (given by the magnitude of the largest eigenvalue,  $c = \|\lambda_{\max}\|$  after multiplying  $A$  by a constant scalar), and the fraction  $d$  of nodes selected as non-driver nodes (Fig. 2e–g). The error remains below approximately 5% for scaling  $c < 1.5$  and non-driver fraction  $d < 0.4$  (Fig. 2e–g). In this paper, we will use these connectomes scaled such that  $c = \|\lambda_{\max}\| = 1$ , and non-driver fraction  $d \leq 0.4$ , to ensure generalizability of our findings to the non-simplified versions of these same networks.

## 2.4. Determinant of the Driver-to-Non-Driver Network

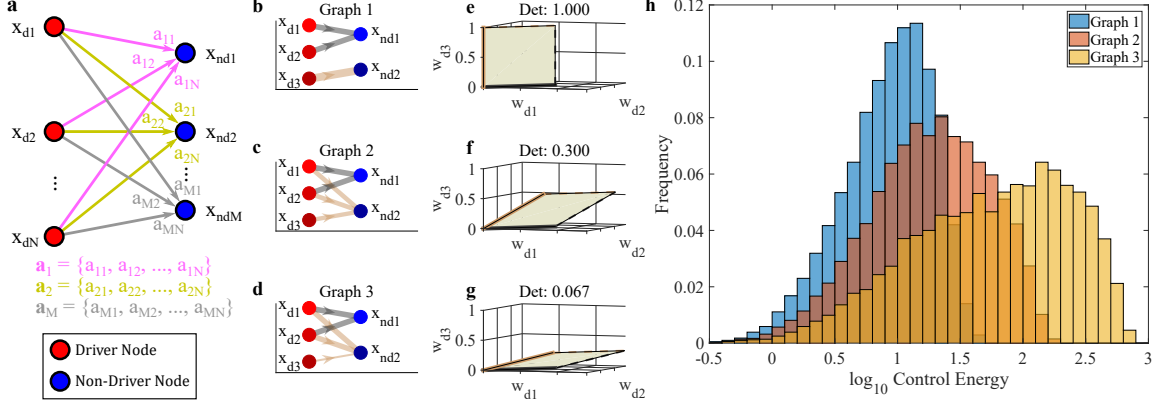


Figure 3: **Geometric Intuition & Control Energies of First-Order Networks.** (a) Graph representation of a simplified first-order network containing connections from  $N$  driver nodes in red to  $M$  non-driver nodes in blue. The edges connecting all driver nodes to the  $i$ -th non-driver corresponding to the  $i$ -th row of  $A_{21}$  are shown in different colors. (b) Graph representation of a network with driver nodes in red, non-driver nodes in blue, weight distribution into non-driver 1 in gray, and weight distribution into non-driver 2 in tan, for dissimilarly distributed weights, (c) for somewhat similarly distributed weights, and (d) for very similarly distributed weights. (e) Geometric representation of the parallelotope formed by the 2 vectors of weight distributions into non-drivers 1 and 2, with the volume shaded in beige for dissimilarly distributed weights, (f) for somewhat similarly distributed weights, and (g) for very similarly distributed weights. (h) Base-10 log distribution of control energy required to bring each graph to 10,000 random final states  $\mathbf{x}_{nd}^* \in (-1, 1)^M, \mathbf{x}_d^* \in (-1, 1)^N$ .

After deriving a closed-form approximation for the minimal energy to control a network, we seek a physical interpretation of the mathematical features that predict the control energy.

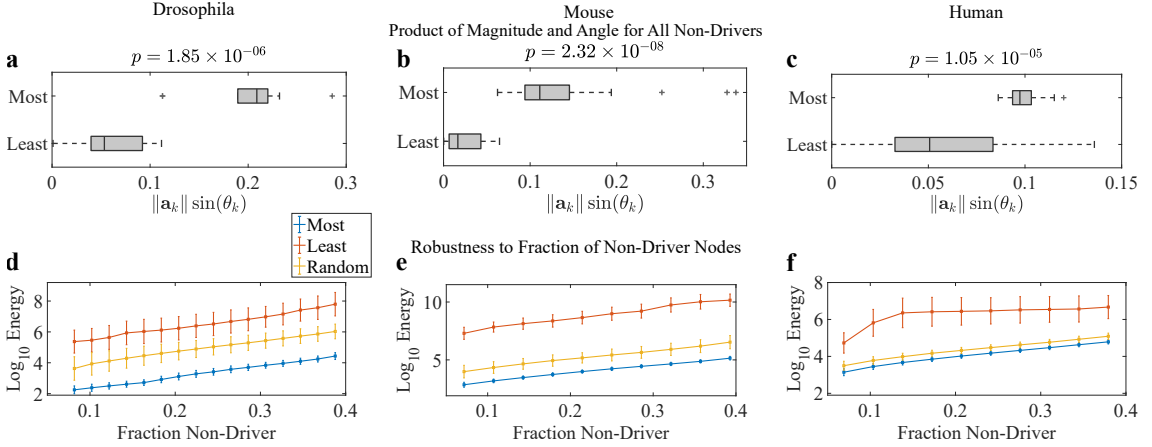
We let  $Q = A_{21}A_{21}^T$ , and write Eq. (2.2) as

$$E(\mathbf{u}) = 12 \frac{\mathbf{v}_1^T \text{adj}(Q) \mathbf{v}_1}{\det(Q)} + \mathbf{v}_2^T \mathbf{v}_2, \quad (2.3)$$

where  $\mathbf{v}_1 = \mathbf{x}_{nd}^* - \frac{1}{2}A_{21}\mathbf{x}_d^*$  and  $\mathbf{v}_2 = \mathbf{x}_d^*$ , and  $\text{adj}(Q)$  is the adjugate matrix of  $Q$ . We notice that the determinant of  $Q$  acts as a scaling factor for the total energy. This insight is useful because of the geometric interpretation of a Gram matrix determinant. Specifically, let  $\mathbf{a}_i \in \mathbb{R}^{1 \times N}$  be the  $i$ -th row of  $A_{21}$  (which we will call the *weight vector*), representing weights from all  $N$  drivers to the  $i$ -th non-driver node (Fig. 3a). Then, the determinant of the Gram matrix  $Q$  is equal to the squared volume of the parallelotope formed by all  $\mathbf{a}_i$ .

To gain an intuition for these results, we show a simple system with 3 drivers and 2 non-drivers with varying network topologies in Fig. 3b–d, and their corresponding geometric parallelotopes in Fig. 3e–g with weight-vector  $\mathbf{a}_1$  in gray and  $\mathbf{a}_2$  in tan. We also compute the distribution of control energy required to drive each network from initial states  $\mathbf{x}_d = 0$ ,  $\mathbf{x}_{nd} = 0$  to 10,000 random final states  $\mathbf{x}_{nd}^* \in (-1, 1)^M$ ,  $\mathbf{x}_d^* \in (-1, 1)^N$  in Fig. 3h. As the non-drivers  $x_{nd1}, x_{nd2}$  become more similarly connected, the total area of the parallelotope (and corresponding Gram determinant) decreases (Fig. 3e–g), and the control energy increases (Fig. 3h). We note that this determinant relationship persists for any number of nodes where  $N > M$ . We conclude that the similarity between weight-vectors generally scales the control energy through  $\det(Q)$ , allowing us to analyze and modify the connectivity of a network with respect to its control energy.

## 2.5. Identifying Energetically Favorable Control Nodes



**Figure 4: Topological Characteristics & Energetic Performance of Networks.** (a) Boxplot of each non-driver weight-vector’s magnitude and angle product ( $\|\mathbf{a}_k\| \sin(\theta_k)$ ) between the energetically most and least favorable networks in the Drosophila, (b) mouse, and (c) human connectomes, for a non-driver fraction of 0.2 and  $p$ -values from a 2-sample  $t$ -test. (d) Mean and standard deviations of the base-10 log of the minimum control energies required to bring the system to 2000 random final states  $\mathbf{x}_{nd}^* \in (-1, 1)^M$ ,  $\mathbf{x}_d^* \in (-1, 1)^N$  for each of a range of non-driver fractions for the energetically most favorable, least favorable, and random networks for the Drosophila, (e) mouse, and (f) human.

Here, we further explore the idea of “similarity” between connections  $\mathbf{a}_i$ , to quantify the

impact of each individual non-driver on the control energy.

### 2.5.1. Topological Contributors to Control Energy.

Our analysis is rooted in the intuition that the edge weights  $\mathbf{a}_i$  that maximize the parallelotope volume, thereby facilitating network control, are large in magnitude and orthogonal to each other. Let  $\lambda_i$  and  $\mathbf{e}_i$  be the eigenvalues and eigenvectors of the matrix  $Q$  in Eq. (2.3). We derive in Lemma X.6 the equivalent, alternative control energy expression

$$E(\mathbf{u}) = 12 \left( \frac{\sum_{i=1}^M w_i c_i^2}{\sum_{i=1}^M w_i} \right) \left( \sum_{k=1}^M \frac{1}{\|\mathbf{a}_k\|^2 \sin(\theta_k)^2} \right) + \mathbf{v}_2^T \mathbf{v}_2, \quad (2.4)$$

where  $w_i = \prod_{j \neq i}^M \lambda_j$ ,  $c_i = \mathbf{e}_i^T \mathbf{v}_1$ , and  $\theta_k$  is the angle formed between  $\mathbf{a}_k$  and the parallelotope formed by  $\mathbf{a}_{j \neq k}$ . We also derive in Lemma X.7 the average control energy to reach all random final states drawn uniformly from -1 to 1,  $\mathbf{x}_{\text{nd}}^* \in (-1, 1)^M$ ,  $\mathbf{x}_{\text{d}}^* \in (-1, 1)^N$ , as

$$\mathbb{E}[E(\mathbf{u})] = \frac{1}{3}N + M + 4 \left( \sum_{k=1}^M \frac{1}{\|\mathbf{a}_k\|^2 \sin(\theta_k)^2} \right). \quad (2.5)$$

For  $N$  drivers and  $M$  non-drivers, we can visualize the  $M$  weight vectors  $\mathbf{a}_k$  as forming a parallelotope in an  $N$ -dimensional space. The variable  $\theta_k$  then represents the angle formed between  $\mathbf{a}_k$  and the parallelotope formed by the remaining  $M-1$  vectors  $\mathbf{a}_{j \neq k}$ . An example with  $N = 3, M = 2$  is shown in Fig. 3e–g, where  $\theta_1 = \theta_2$  is the angle between the tan and gray vectors.

Here, we have segregated the control energy into a task-based  $\left( \frac{\sum_{i=1}^M w_i c_i^2}{\sum_{i=1}^M w_i} \right)$  and topology-based  $\left( \sum_{k=1}^M \frac{1}{\|\mathbf{a}_k\|^2 \sin(\theta_k)^2} \right)$  term (Eq. 2.4), where the average minimum control energy depends linearly on the topology-based term (Eq. 2.5). This segregation allows us to analyze the topology separate from the specific control task, and shows that each non-driver additively contributes to the total control energy minimally when  $\|\mathbf{a}_i\|$  and  $\sin(\theta_i)$  are large.

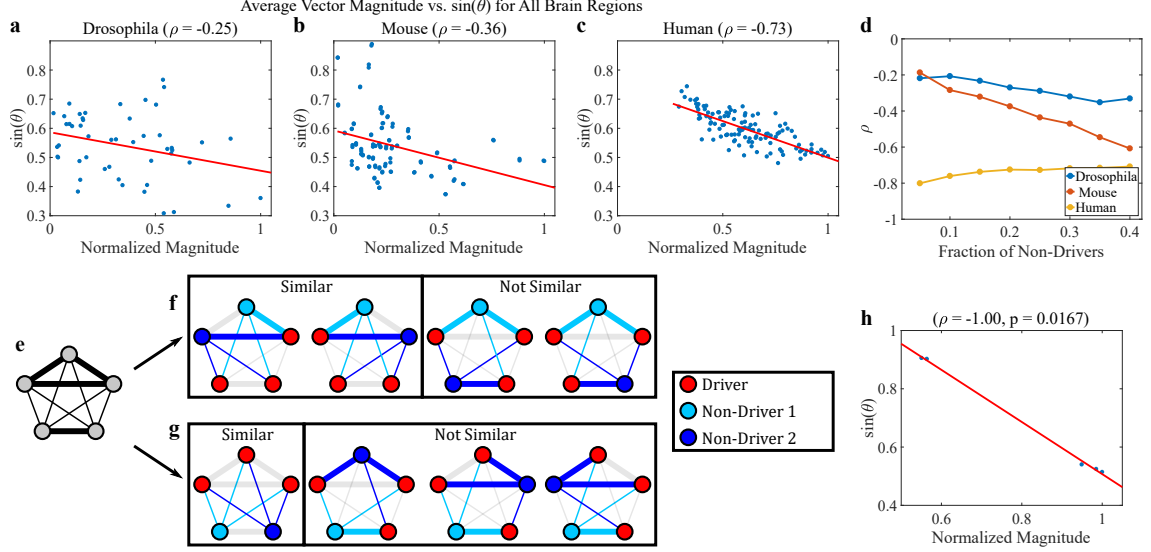
### 2.5.2. Energetically Favorable Driver-Non-Driver sets.

To support this discussion, we used expression Eq. (2.4) to find the selections of  $M$  non-drivers that minimized and maximized this topology term (see appendix in Chapter 3), which we define as the *energetically most favorable* and *energetically least favorable* selections, respectively. We show example distributions of each weight-vector's magnitude  $\|\mathbf{a}_k\|$  times angle  $\sin(\theta_k)$  (Fig. 4a–c) between these selections in *Drosophila*, mouse, and human for non-driver fraction 0.2. We observe that the energetically least favorable selections have significantly weaker magnitudes and angles than the most favorable selections.

Next, we demonstrate the utility and robustness of these topological features for control by computing the minimum control energy along the non-simplified networks using the driver and non-driver designations from the simplified networks in Eq. (2.4) for a range of non-driver fractions. For each non-driver fraction and species, we computed the control energy to bring the energetically most and least favorable non-driver selections, and 2000 random non-driver selections to a corresponding set of 2000 random final states  $\mathbf{x}_{\text{nd}}^* \in (-1, 1)^M$ ,  $\mathbf{x}_{\text{d}}^* \in (-1, 1)^N$  (Fig. 4j–l). Across all three species, the most favorable selections require around 0.5–1 order of magnitude less control energy than the random selections, and 2.5–4 orders of magnitude less control energy than the least favorable selections. This difference indicates an energetic advantage for some configurations of drivers and non-drivers over others.

## 2.6. Complex Brain Networks are Energetically Favorable

Given the relationship between a network's connectivity and minimum control energy in Eq. (2.4), we seek to understand if brain networks are organized along energetically favorable principles. Fundamentally, we ask how well a network's specific set of connectivity features  $\|\mathbf{a}_k\|$  and  $\sin(\theta_k)$  combine to minimize the topology-dependent energy term  $\sum_{k=1}^M \frac{1}{\|\mathbf{a}_k\|^2 \sin(\theta_k)^2}$ . In networks that are not designed along these energetic principles, we expect to see no particular relationship between  $\|\mathbf{a}_k\|$  and  $\sin(\theta_k)$ . In networks that minimize the topology dependent energy term, we expect a compensatory effect, where non-drivers



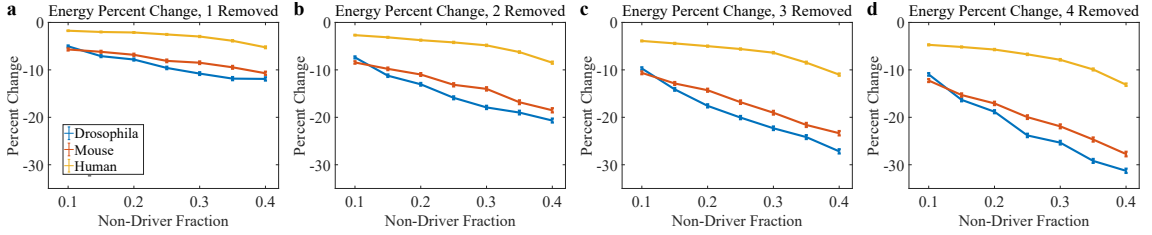
**Figure 5: Energetically Favorable Organization of Network Topology.** (a) Average  $\sin(\theta_k)$  versus normalized  $\|\mathbf{a}_k\|$  for each brain region across 10,000 random non-driver selections for a non-driver fraction of 0.2, along with best fit line (red) and corresponding Spearman correlation coefficient in the Drosophila, (b) mouse, and (c) human. (d) Spearman correlation coefficients in the Drosophila, mouse, and human over 2,000 random non-driver selections for each of a range of non-driver fractions. (e) Example toy network of 5 nodes with three strongly interconnected nodes at the top, and two strongly interconnected nodes at the bottom. (f) Representation of similarity in driver  $\rightarrow$  non-driver connections between Non-Driver 1 (light blue, member of three strongly connected nodes) and all possible selections of Non-Driver 2 (blue). Across all 4 configurations, Non-Driver 1 has an average of 1.5 strong connections, and 2/4 similarly connected (small angle) configurations. (g) Similarity in driver  $\rightarrow$  non-driver connections between Non-Driver 1 (light blue, member of two strongly connected nodes) and all selections of Non-Driver 2 (blue). Across all 4 configurations, Non-Driver 1 has an average of 0.75 strong connections, and 1/4 similarly connected configurations. (h) Plot of average magnitude versus  $\sin(\theta)$  for the toy network, with Spearman rank correlation coefficient.

with small angles have large magnitudes, and *vice versa*.

To explore the relationship between  $\|\mathbf{a}_k\|$  and  $\sin(\theta_k)$  in brain networks, we selected 10,000 random permutations of non-drivers in each of the Drosophila, mouse, and 10 human connectomes, at non-driver fraction  $d$ . For each permutation, we calculated  $\|\mathbf{a}_k\|$  and  $\sin(\theta_k)$  for every non-driver. Then, we averaged  $\|\mathbf{a}_k\|$  and  $\sin(\theta_k)$  for each non-driver across all permutations, giving us an averaged magnitude  $\|\mathbf{a}_k\|$  and  $\sin(\theta_k)$  for each brain region in each network. Finally, we plotted the averaged  $\sin(\theta_k)$  versus  $\|\mathbf{a}_k\|$  for all brain regions

in each network for  $d = 0.2$  (Fig. 5a–c). We find little relationship between the averaged  $\|\mathbf{a}_k\|$  and  $\sin(\theta_k)$  in the Drosophila (Spearman  $\rho = -0.25$ ,  $p = 0.0748$ ), a moderate negative relationship in the mouse ( $\rho = -0.36$ ,  $p = 0.000125$ ), and a strong negative relationship in the human ( $\rho = -0.73$ ,  $p \approx 0$ ). This ordering holds for a wide range of non-driver fractions (Fig. 5d). We graphically demonstrate how this negative  $\sin(\theta_k)$  *versus*  $\|\mathbf{a}_k\|$  relation might arise in networks, using a simple 5-node network with two communities of 3 and 2 strongly interconnected sets of nodes (Fig. 5d–f), which has a strong negative relationship (Fig. 5h).

## 2.7. Network Manipulation to Facilitate Control



**Figure 6: Reducing Minimum Control Energy Through Edge Deletion.** (a) Means and standard errors of percent change in control energy before and after deleting edges that maximally increase the determinant based on Eq. (2.6) over 2,000 control tasks, with initial states  $\mathbf{x}_{\text{nd}}(0) = 0$ ,  $\mathbf{x}_{\text{d}}(0) = 0$ , and random final states  $\mathbf{x}_{\text{nd}}^* \in (-1, 1)^M$ ,  $\mathbf{x}_{\text{d}}^* \in (-1, 1)^N$ . Non-drivers were randomly selected for a range of non-driver fractions in the Drosophila, mouse, and human connectomes for 1 deletion, (b) 2 deletions, (c) 3 deletions, and (d) 4 deletions. Standard errors were computed as  $SE = \frac{s}{\sqrt{n}}$ , where  $s$  is the sample standard deviation over the 2,000 tasks, and  $n = 2,000$ .

Here, we consider network modifications that lead to lower control energies. We focus on the effects of edge deletion since it is often useful in the study of biological systems such as brain (Alstott et al., 2009), metabolic (Aristidou et al., 1994), and gene regulatory (Sander and Joung, 2014) networks. Specifically, we quantify the effect of modifying each edge weight on the determinant in Lemma X.5 as

$$\frac{\partial}{\partial A_{21}} \det(Q) = 2 \det(Q) (Q^{-1} A_{21}), \quad (2.6)$$

and compute the decrease in control energy as a result of deleting edges that maximally



increase the determinant.

First, for each species and each of a range of non-driver fractions, we randomly selected 2,000 permutations of non-drivers. For each permutation, we extracted the block matrix  $A_{21}$ , calculated  $2\det(Q)(Q^{-1})A_{21}$ , and found the element  $a_{ij} \neq 0$  yielding the largest increase in  $\det(Q)$  based on Eq. (2.6). We then simulated an edge deletion by setting  $a_{ij} = 0$ , and repeated the process to obtain networks of 1, 2, 3, and 4 deleted edges. Finally, we computed the percent change in control energy required to bring the non-simplified network from initial states  $\mathbf{x}_{\text{nd}}(0) = 0$ ,  $\mathbf{x}_{\text{d}}(0) = 0$ , to final states  $\mathbf{x}_{\text{nd}}^* \in (-1, 1)^M$ ,  $\mathbf{x}_{\text{d}}^* \in (-1, 1)^N$  before and after edge deletion (Fig. 6a–d).

As can be seen in Fig. 6a, the removal of one edge can sometimes lead to more than a 10% average reduction in control energy, while the removal of four edges (Fig. 6d) can sometimes lead to more than a 30% reduction. Across most non-driver fractions, the *Drosophila* experienced greater energy reduction than the mouse, which also experienced greater energy reduction than the human. This corresponds to the previous finding where, because brain networks of these increasingly complex species are already energetically favorably wired, they may not experience as much improvement after modification.

## 2.8. Contribution and Future Directions

The control of networked systems is a critical frontier in science, mathematics, and engineering, as it requires a fundamental understanding of the mechanisms that drive network dynamics and subsequently offers the knowledge necessary to intervene in real-world systems to better their outcomes (Motter, 2015). While some theoretical predictions exist in nonlinear network systems (Cornelius et al., 2013a), the majority of recent advances have been made in the context of linear control (Liu et al., 2011; Ruths and Ruths, 2014). Nevertheless, basic intuitions regarding how edge weights impact control have remained elusive. Although spectral analysis of a network’s controllability Gramian (Kailath, 1980) yields theoretically useful information about the overall behavior of the network under con-

trol (Pasqualetti et al., 2014), it is not obvious how specific patterns of connectivity or selections of driver and non-driver nodes contribute to this behavior. Understanding this relationship is crucial when analyzing empirical biological networks such as the brain, where nodes and edges often have known functions (Lanteaume et al., 2007) that may modulate or influence one other.

A distinct advantage of our approach is the focus on a physically meaningful topological understanding of the principles governing network control. We map control behavior to network topology through a simplified network only involving connections from driver to non-driver nodes. This simplification hard-codes the fact that energy can be transmitted directly from drivers to non-drivers along walks of length unity, and is motivated by recent work demonstrating that relatively sparse network representations of complex biological systems (Park et al., 2015; Liu et al., 2016) can contain much of the information needed to understand the system’s structure and dynamics (Clauset et al., 2008; Zhu and Xia, 2015). Our results inform our understanding of how much first-order connections contribute to the overall dynamics of our network control systems. Moreover, they inform the development of analytical constraints on the accessible state space of a networked system, particularly informing the set of states within which one might seek to push the brain using stimulation paradigms common in the treatment of neurological disorders and psychiatric disease (Chen et al., 2014; Chrysikou and Hamilton, 2011). While many initial studies have examined unconstrained state spaces (Gu et al., 2015; Betzel et al., 2016a; Muldoon et al., 2016), understanding viable states and state trajectories is critical for the translation of these ideas into the clinic (Bassett et al., 2017). Further, by formally quantifying the contribution of the network connectivity to the control energy, we lay the groundwork for the optimization of stimulation sites in neural systems, a problem that has received very little theoretical treatment, and is considered one of the current critical challenges in neuroengineering (Johnson et al., 2013).

Finally, we make strategic, task-agnostic edge deletions that maximally increase the de-

terminant and observe that, even in an overdetermined, unsimplified system ( $N > M$ ), a single edge deletion could produce a profound improvement in the general controllability of a network. This sensitivity suggests that dynamical networks such as the brain can produce fairly drastic changes in dynamical behavior given minute changes in physiological topology, consistent with observations of critical dynamics in human and animal neurophysiology (Rubinov et al., 2011; Shew et al., 2015). Moreover, these results also suggest that minor, targeted structural changes through concussive injury can lead to drastic changes in overall brain function (Caeyenberghs et al., 2016; van der Horn et al., 2016), via altering the controllability landscape of the brain (Gu et al., 2017). We further observed that these topological modifications were task-agnostic edge deletions, signifying that even in a linear regime, the presence of an unfavorable edge can have a profoundly negative impact on the controllability of a network. We note that it is natural to perform a similar analysis that takes into account the specific tasks  $\mathbf{v}_1, \mathbf{v}_2$  by taking the derivative of the full energy term  $E_{total}$  with respect to  $A_{21}$ , which would optimize the network topology for a specific task, as studied in more detail in (Betzel et al., 2016a).

To achieve the most meaningful comparison between species, we only analyzed weighted meso-scale whole brain networks. As such, we did not include binary neuronal connectomes (e.g., *C. elegans*), and binary or partial connectomes (e.g., macaque). As more connectomes become available, we hope to further explore the role of species complexity on network controllability. Until then, we consider the comparison of energetically favorable connectivity between species to be a preliminary excursion into a nuanced evolutionary phenomena. As demonstrated in the significant percent change in energy after edge deletion, we emphasize that uncertainty in network connectivity has the potential to yield substantial changes in average control energy. Finally, we note that while methodological limitations prevent us from resolving excitatory *versus* inhibitory connectivity, all results are directly applicable to networks with signed elements. Further important theoretical considerations and methodological limitations pertinent to our approach, linear model of dynamics, optimality of control trajectories, and empirical data sets are discussed in the appendix in Chapter 3.

## 2.9. Conclusion

In closing, we note that the natural direction in which to take this work will be to use higher-order approximations of this framework found in the appendix in Chapter 3 to gain intuition for the role of complex network topologies (e.g. self-loops, cycles) in controlling networks. Moreover, it would be interesting to apply this reduced framework to random graphs and other well-known benchmarks – both from a mathematical perspective (Bollobas, 1985) and also in the context of neural systems (Klimm et al., 2014; Sizemore et al., 2016) – to better understand the phenotypes present in those graph ensembles. Third and finally, informing the design of new networks with these tools may be particularly useful in neuromorphic computing (Pfeil et al., 2013), material science (Giusti et al., 2016), and other contexts where optimal control of physical systems is of paramount importance.

## CHAPTER 3 : Appendix to Role of Graph Architecture in Controlling Dynamical Networks

### 3.1. Connectome Data

*Drosophila Connectome.* The full reconstruction of the *Drosophila* connectome can be found in the FlyCircuit 1.1 database (Chiang et al., 2011; Shih et al., 2015). This database contains images of 12,995 neurons, as well as their projections, that are characteristic of the *Drosophila* female. In this database, each neuron was labeled using green fluorescent protein (GFP) and its location was estimated from 3-dimensional images that were co-registered to a template using a rigid linear transform. To obtain a mesoscale representation of this fine-scale data, neurons were assigned to one of 49 local populations, based on their morphology and known functions. Following the original work from Shih and colleagues, we treated each of these 49 populations as the nodes of the network, and we treated the directed, weighted edges between populations as network edges (Shih et al., 2015).

*Mouse Inter-Areal Connectome Data.* In addition to the *Drosophila* connectome, we also analyzed the inter-areal connectome of the mouse. In particular, we use the exact network studied in (Rubinov et al., 2015), which was reconstructed from original tract-tracing data recently released by the Allen Brain Institute (Oh et al., 2014). The entire brain was separated into 112 regions, which we treat as network nodes. Each pair of regions was then linked by directed edges that encoded the presence or absence of inter-regional projections. The weight of each edge was defined by the number of projections normalized by the volumes of the two regions being connected.

*Human Diffusion Imaging Data.* Ten healthy adult human subjects (m) were imaged as part of an ongoing data collection effort at the University of Pennsylvania; the subjects provided informed consent in writing, in accordance with the Institutional Review Board of the University of Pennsylvania. All scans were acquired on a Siemens Magnetom Prisma 3 Tesla scanner with a 64-channel head/neck array at the University of Pennsylvania. Each

data acquisition session included both a diffusion spectrum imaging (DSI) scan as well as a high-resolution T1-weighted anatomical scan. The diffusion scan was 730-directional with a maximum  $b$ -value of 5010s/mm<sup>2</sup> and TE/TR = 102/4300 ms, which included 21  $b = 0$  images. Matrix size was 144×144 with a slice number of 87. Field of view was 260×260mm<sup>2</sup> and slice thickness was 1.80mm. Acquisition time per DTI scan was 53:24min, using a multi-band acceleration factor of 3. The anatomical scan was a high-resolution three-dimensional T1-weighted sagittal whole-brain image using a magnetization prepared rapid acquisition gradient-echo (MPRAGE) sequence. It was acquired with TR = 2500 ms; TE=2.18 ms; flip angle = 7 degrees; 208 slices; 0.9mm thickness.

DWI is highly sensitive to subject movement (Yendiki et al., 2013), which can cause significant distortions in the reconstructed ODFs if not corrected. Motion correction is typically applied by determining an affine or non-linear transform to align each DWI volume to a reference derived from the high-signal  $b = 0$  images. The high  $b$ -values used in DSI present a problem for this approach, as the low signal in many of the volumes leads to poor registration. To address this issue, we interspersed  $b = 0$  volumes in the scan sequence, one for every 35 volumes. An initial average template was produced by averaging the  $b = 0$  images together and then improved by registering the  $b = 0$  images to the initial template and re-averaging. Each  $b = 0$  was finally re-registered to the improved template, and then each volume in the DSI scan was then motion corrected by applying the transformation calculated for the closest  $b = 0$  volume. Motion correction also impacts the effective  $b$ -matrix directions since the rotated images are no longer aligned with the scanner; therefore the transforms applied to motion correct each volume were also used to rotate the corresponding  $b$ -vectors (Leemans and Jones, 2009). The processing pipeline was implemented using Nipype (Gorgolewski et al., 2011) with registration performed using the Advanced Normalization Tools (ANTs) (Avants et al., 2011).

Using DSI-Studio (<http://dsi-studio.labsolver.org>), orientation density functions (ODFs) within each voxel were reconstructed from the corrected scans using GQI (Yeh et al., 2010).

We then used the reconstructed ODFs to perform a whole-brain deterministic tractography using the derived QA values in DSI-Studio (Yeh et al., 2013). We generated 1,000,000 streamlines per subject, with a maximum turning angle of 35 degrees (Bassett et al., 2011) and a maximum length of 500mm (Cieslak and Grafton, 2014).

We constructed networks for each subject where nodes are atlas regions and edges are the measured connection strength between region pairs (Hagmann et al., 2008). The nodes of the network were derived from spatially-defined regions of a brain atlas. We chose the anatomically-defined AAL atlas, originally developed in Statistical Parametric Mapping (SPM) (Tzourio-Mazoyer et al., 2002), which divides each brain hemisphere into 45 regions, and then includes cerebellar regions as well. We used a version in MNI-space that was then warped into subject-specific space using ANTs. Edges of the network were constrained to the set of streamlines that both started and terminated within a pair of regions. The mean QA weighting across streamlines connecting the two regions was used as an estimate of the strength of the connection in order to examine individual variability in structural connectivity (Griffa et al., 2013).

### 3.2. Selection of Energetically Most & Least Favorable Non-Drivers

In result section C, we use the minimum energy expression Eq. 2.4 to select the  $N$  driver and  $M$  non-driver nodes that maximized and minimized the topology-based term. Due to the immense number of possible permutations (e.g. over  $10^{24}$  ways to select 23 non-drivers in the 116 node human connectome), we made use of a greedy algorithm.

For each connectome, we first computed the topology-based term for all possible selections of  $M = 4$  non-drivers ( $\approx 2 \times 10^5$  for *Drosophila*,  $\approx 6 \times 10^6$  for mouse, and  $\approx 7 \times 10^6$  for human), and found the selections that maximized and minimized the topology-based term. Then, to reach  $M = 5$  non-drivers, we selected the  $N$  remaining driver nodes one at a time to be a non-driver node, and found the node that maximized (or minimized) the topology-based term. We continued this greedy selection for  $M = 6, 7, \dots$  to reach the

desired non-driver fraction. To ensure that our selection was fully *reachable* (i.e. the Gram matrix  $A_{21}A_{21}^T$  had full rank), we computed the rank of the Gram matrix  $A_{21}A_{21}^T$ , and only selected nodes among those that maintained full rank.

### 3.3. First-Order Energy Approximation

Here we reiterate our mathematical notation and assumptions in greater detail, and provide lemmas for the main results. Consider a network represented by the directed graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ , where  $\mathcal{V} = \{1, \dots, n\}$  and  $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$  are the sets of network vertices and edges, respectively. Let  $a_{ij} \in \mathbb{R}$  be the weight associated with the edge  $(i, j) \in \mathcal{E}$ , and let  $A = [a_{ij}]$  be the weighted adjacency matrix of  $\mathcal{G}$ . We associate a real value (*state*) with each node, collect the nodes' states into a vector (*network state*), and define the map  $\mathbf{x} : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^n$  to describe the evolution (*dynamics*) of the network state over time. We let the network dynamics be linear and time invariant, as described by the equation

$$\dot{\mathbf{x}} = A\mathbf{x}. \quad (3.1)$$

We are particularly interested in characterizing how the network structure  $\mathcal{G}$  influences the control properties of the dynamical system (3.1). We assume that a subset of  $N$  nodes, called drivers, is independently manipulated by external controls and, without loss of generality, we reorder the network nodes such that the  $N$  drivers come first. Thus, the network dynamics with controlled drivers read as

$$\begin{bmatrix} \dot{\mathbf{x}}_{\text{d}} \\ \dot{\mathbf{x}}_{\text{nd}} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{\text{d}} \\ \mathbf{x}_{\text{nd}} \end{bmatrix} + \begin{bmatrix} I_N \\ 0 \end{bmatrix} \mathbf{u}, \quad (3.2)$$

where  $\mathbf{x}_{\text{d}}$  and  $\mathbf{x}_{\text{nd}}$  are the state vectors of the driver and non-driver nodes,  $A_{11} \in \mathbb{R}^{N \times N}$ ,  $M = n - N$ ,  $A_{12} \in \mathbb{R}^{N \times M}$ ,  $A_{21} \in \mathbb{R}^{M \times N}$ ,  $A_{22} \in \mathbb{R}^{M \times M}$ ,  $I_N$  is the  $N$ -dimensional identity matrix, and  $\mathbf{u} : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^N$  is the control input.



We will use the word *controllable* to refer to networks that are *point-to-point controllable* at time  $T \in \mathbb{R}_{\geq 0}$  if, for any pair of states  $\mathbf{x}_d^*$  and  $\mathbf{x}_{nd}^*$ , there exists a control input  $u$  for the dynamics (2.1) such that  $\mathbf{x}_d(T) = \mathbf{x}_d^*$  and  $\mathbf{x}_{nd}(T) = \mathbf{x}_{nd}^*$ . We refer the interested reader to (Kailath, 1980) for a detailed discussion and rigorous conditions for the controllability of a system with linear dynamics. Finally, we define the energy of  $u$  as

$$E(\mathbf{u}) = \sum_{i=1}^N \underbrace{\int_0^T u_i(t)^2 dt}_{E_i},$$

where  $u_i$  is the  $i$ -th component of  $\mathbf{u}$ . In what follows we characterize how the network topology and weights determine the control energy needed for a given control task. We restrict our analysis to a class of bipartite networks, as specified in the following assumptions. We remark that our assumptions, although restrictive, allow us to thoroughly predict how driver  $\rightarrow$  non-driver connections facilitate or inhibit network control even in more complex network models (see Section 3.5). We later relax some of the assumptions to extend our approximation to networks that are neither bipartite nor feed-forward (see Section 3.4).

**Assumption 1.** *The initial state of the network satisfies  $\mathbf{x}_d(0) = 0$  and  $\mathbf{x}_{nd}(0) = 0$ .*  $\square$

**Assumption 2.** *The network  $\mathcal{G}$  contains only edges from the drivers to the non-drivers, that is,  $A_{11} = 0$ ,  $A_{22} = 0$ , and  $A_{12} = 0$ . Thus, the dynamics (2.1) simplify to  $\dot{\mathbf{x}}_d = \mathbf{u}(t)$ , and  $\dot{\mathbf{x}}_{nd} = A_{21}\mathbf{x}_d$ .*  $\square$

From assumptions 1 and 2 we readily observe that

$$\begin{aligned} \mathbf{x}_d(t) &= \int_0^t \mathbf{u}(\tau) d\tau, \text{ and} \\ \mathbf{x}_{nd}(t) &= A_{21} \int_0^t \mathbf{x}_d(\tau) d\tau = A_{21} \int_0^t \int_0^\tau \mathbf{u}(\tau_1) d\tau_1 d\tau. \end{aligned} \tag{3.3}$$

We see that  $\mathbf{x}_{nd}$  provides an integral constraint to  $\mathbf{x}_d$ , and we represent the specific values

of the constraints as

$$\int_0^t \mathbf{x}_d(\tau) d\tau = \mathbf{C}, \quad (3.4)$$

where  $\mathbf{C} \in \mathbb{R}^{N \times 1}$  is an  $N$ -dimensional vector of real-valued constants. Furthermore, the set of controllable states can be characterized as follows. For a matrix  $M$ , let  $\text{Im}(M)$  and  $\text{Rank}(M)$  denote the image and rank of  $M$ , respectively (Meyer, 2001).

**Lemma 3.3.1. (*Controllability*)** *The network (2.1) is controllable if and only if the rank  $\text{Rank}(A_{21}) = M$ . Furthermore, the set of controllable states is  $\text{Im} \left( \begin{bmatrix} I_N & 0 \\ 0 & A_{21} \end{bmatrix} \right)$ .*

*Proof.* Notice that the controllability matrix of (2.1) is

$$\mathcal{C} = \begin{bmatrix} B & AB \end{bmatrix} = \begin{bmatrix} I_N & 0 \\ 0 & A_{21} \end{bmatrix},$$

and recall that a state is controllable if and only if it belongs to the range space of the controllability matrix. □

**Lemma 3.3.2. (*Minimum Energy Control Input*)** *The  $i^{\text{th}}$  driver trajectory  $x_{di}(t)$  that minimizes the control energy takes the form  $x_{di}(t) = a_i t^2 + b_i t$*

*Proof.* Recall from (2.1) and assumption 2 that  $\dot{x}_{di}(t) = u_i(t)$ . We minimize the energy

$$\begin{aligned} E_i &= \min_{u_i} \int_0^T u_i(t)^2 dt \\ &= \min_{x_{di}} \int_0^T \dot{x}_{di}(t)^2 dt, \end{aligned}$$

where  $x_{di}(0) = 0, x_{di}(T) = x_{di}^*$ . From (3.4),  $x_{di}$  is also subject to some integral constraint

$$\int_0^T x_{di}(t)dt = C_i,$$

where  $C_i$  is the  $i$ th element of  $\mathbf{C}$ . We see this naturally takes the form of the isoperimetric problem in the calculus of variations, which finds

$$\min_{x_{di}} \int_a^b F(t, x_{di}, \dot{x}_{di})dt,$$

where  $F(t, x_{di}, \dot{x}_{di}) = \dot{x}_{di}(t)^2$ ,  $a = 0$ , and  $b = T = 1$ , constrained by

$$\int_a^b G(t, x_{di}, \dot{x}_{di})dt = 0,$$

where  $G(t, x_{di}, \dot{x}_{di}) = x_{di}(t) - C_i$ . The trajectory  $x_{di}^*(t)$  which locally minimizes the cost function must satisfy the necessary (Euler-Lagrange) and sufficient (Jacobi) conditions. The Euler-Lagrange equation reads

$$\frac{d}{dt} \frac{\partial}{\partial \dot{x}} (F + \lambda G) = \frac{\partial}{\partial x} (F + \lambda G),$$

which, after substituting  $F$  and  $G$ , yields

$$\ddot{x}_{di}^*(t) = \frac{\lambda}{2},$$

to give the only extremal solution satisfying assumption 1

$$x_{di}^*(t) = \frac{\lambda}{2}t^2 + bt,$$

where  $\lambda$  is the lagrange multiplier. Because  $(F + \lambda G)_{xx} = (F + \lambda G)_{x\dot{x}} = 0$ , and the quantity

$(F + \lambda G)_{\dot{x}\dot{x}} = 2$ , the Jacobi condition becomes

$$\int_0^1 \dot{\eta}^2 (F + \lambda G)_{\dot{x}\dot{x}} dt = 2 \int_0^1 \dot{\eta}^2 dt \geq 0,$$

which holds true for any arbitrary smooth function  $\eta$ , where  $\eta(0) = \eta(1) = 0$ . As  $x_{\text{di}}^*(t) = at^2 + bt$  is the only extremal function, and is also minimum,  $x_{\text{di}}$  is the global minimum of the constrained control energy.

□

**Lemma 3.3.3. (*Minimum Control Energy*)** *The required control energy for the  $i^{\text{th}}$  driver is  $E_i = 12C_i^2 - 12C_i x_{\text{di}} + 4x_{\text{di}}$ .*

*Proof.* We recall that the energy required to drive  $x_{\text{di}}$  is

$$\begin{aligned} E_i &= \min_{x_{\text{di}}} \int_0^{T=1} \dot{x}_{\text{di}}^2(t) dt \\ &= \int_0^1 4a_i t^2 + 4a_i b_i t + b_i^2 dt \\ &= \frac{4}{3}a_i + 2a_i b_i + b_i^2. \end{aligned}$$

We solve for  $a_i$  and  $b_i$  via the final state and integral constraint to yields equations

$$\begin{aligned} x_{\text{di}}(T=1) &= a_i + b_i = x_{\text{di}}^* \\ \int_0^{T=1} x_{\text{di}}(t) dt &= \frac{1}{3}a_i + \frac{1}{2}b_i = C_i, \end{aligned}$$

from which we get

$$a_i = 3x_{\text{di}} - 6C_i$$

$$b_i = 6C_i - 2x_{\text{di}}.$$

Substituting  $a_i$  and  $b_i$  into the equation for  $E_i$ , we get

$$E_i = 12C_i^2 - 12C_i x_{di} + 4x_{di}^2.$$

□

**Lemma 3.3.4. (*Total Control Energy*)** *The total control energy is  $E_{total} = 12\mathbf{v}_1^T Q^{-1} \mathbf{v}_1 + \mathbf{v}_2^T \mathbf{v}_2$ , where  $\mathbf{v}_1 = \mathbf{x}_{nd}^* - \frac{1}{2}A_{21}\mathbf{x}_d^*$ ,  $\mathbf{v}_2 = \mathbf{x}_d^*$ , and  $Q = A_{21}A_{21}^T$*

*Proof.* Here, we use the method of Lagrange multipliers to minimize the total energy  $f(\mathbf{C})$  as a function of  $\mathbf{C}$  given in (3.4), constrained by  $g(\mathbf{C})$  given by (3.3). We can write the total energy as

$$\begin{aligned} f(\mathbf{C}) = E(\mathbf{u}) &= \sum_{i=1}^N E_i \\ &= \sum_{i=1}^N 12C_i^2 - 12C_i x_{di} + 4x_{di}^2 \\ &= 12\mathbf{C}^T \mathbf{C} - 12\mathbf{C}^T \mathbf{x}_d + 4\mathbf{x}_d^T \mathbf{x}_d, \end{aligned}$$

with  $M$  constraining equations, the set of which are given by

$$\begin{aligned} g(\mathbf{C}) &= \mathbf{x}_{nd}^* - A_{21} \int_0^T \mathbf{x}_d(\tau) d\tau \\ &= \mathbf{x}_{nd}^* - A_{21} \mathbf{C} \\ &= 0, \end{aligned}$$

where the  $k^{th}$  constraint  $g_k(\mathbf{C})$  is given by the  $k^{th}$  row of  $g(\mathbf{C})$ . The method of Lagrange

multipliers defines the Lagrangian given by

$$\begin{aligned}
\mathcal{L}(\mathbf{C}, \lambda_1, \dots, \lambda_M) &= f(\mathbf{C}) + \sum_{k=1}^M \lambda_k g_k(\mathbf{C}) \\
&= f(\mathbf{C}) + g(\mathbf{C})^T \boldsymbol{\lambda} \\
&= 12\mathbf{C}^T \mathbf{C} - 12\mathbf{C}^T \mathbf{x}_d^* - \mathbf{C}^T A_{21}^T \boldsymbol{\lambda} + \mathbf{x}_{nd}^* \boldsymbol{\lambda} + 4\mathbf{x}_d^{*T} \mathbf{x}_d^* \\
&= \mathbf{C}^T (12\mathbf{C} - 12\mathbf{x}_d^* - A_{21}^T \boldsymbol{\lambda}) + \mathbf{x}_{nd}^* \boldsymbol{\lambda} + 4\mathbf{x}_d^{*T} \mathbf{x}_d^*,
\end{aligned}$$

where  $\lambda_k$  is the  $k^{th}$  Lagrange multiplier to compose  $\boldsymbol{\lambda} \in \mathbb{R}^{M \times 1}$ , and sets the gradient of the Lagrangian to 0

$$\begin{aligned}
\nabla_{\mathbf{C}} \mathcal{L}(\mathbf{C}, \lambda_1, \dots, \lambda_M) &= 24\mathbf{C} - 12\mathbf{x}_d - A_{21}^T \boldsymbol{\lambda} \\
&= 0,
\end{aligned}$$

which allows us to solve for  $\mathbf{C}$  with respect to  $\boldsymbol{\lambda}$

$$\mathbf{C} = \frac{1}{24} A_{21}^T \boldsymbol{\lambda} + \frac{1}{2} \mathbf{x}_d^*.$$

By substituting  $\mathbf{C}$  into the total energy equation and grouping terms, we get a preliminary formulation of  $E(\mathbf{u})$  with respect to  $\boldsymbol{\lambda}$

$$\begin{aligned}
E(\mathbf{u}) &= \frac{12}{24^2} \boldsymbol{\lambda}^T A_{21} A_{21}^T \boldsymbol{\lambda} + \left( \frac{12}{24} - \frac{12}{24} \right) \mathbf{x}_d^{*T} A_{21}^T \boldsymbol{\lambda} + (3 - 6 + 4) \mathbf{x}_d^{*T} \mathbf{x}_d^* \\
&= \frac{\boldsymbol{\lambda}^T A_{21} A_{21}^T \boldsymbol{\lambda}}{48} + \mathbf{x}_d^{*T} \mathbf{x}_d^*.
\end{aligned}$$

To solve for  $\boldsymbol{\lambda}$ , we substitute the expression for  $\mathbf{C}$  into our constraint equations  $g(\mathbf{C})$  to

yield

$$\begin{aligned}
g(\mathbf{C}) &= \mathbf{x}_{\text{nd}}^* - A_{21}\mathbf{C} \\
&= \mathbf{x}_{\text{nd}}^* - \frac{1}{24}A_{21}A_{21}^T\boldsymbol{\lambda} - \frac{1}{2}A_{21}\mathbf{x}_{\text{d}}^* \\
&= 0,
\end{aligned}$$

and solve for  $\boldsymbol{\lambda}$

$$\boldsymbol{\lambda} = 24(A_{21}A_{21}^T)^{-1}(\mathbf{x}_{\text{nd}}^* - \frac{1}{2}A_{21}\mathbf{x}_{\text{d}}^*).$$

Substituting  $\boldsymbol{\lambda}$  into  $E(\mathbf{u})$ , we get

$$\begin{aligned}
E(\mathbf{u}) &= 12(\mathbf{x}_{\text{nd}}^* - \frac{1}{2}A_{21}\mathbf{x}_{\text{d}}^*)^T(A_{21}A_{21}^T)^{-1}(A_{21}A_{21}^T)(A_{21}A_{21}^T)^{-1}(\mathbf{x}_{\text{nd}}^* - \frac{1}{2}A_{21}\mathbf{x}_{\text{d}}^*) + \mathbf{x}_{\text{d}}^{*T}\mathbf{x}_{\text{d}}^* \\
&= 12(\mathbf{x}_{\text{nd}}^* - \frac{1}{2}A_{21}\mathbf{x}_{\text{d}}^*)^T(A_{21}A_{21}^T)^{-1}(\mathbf{x}_{\text{nd}}^* - \frac{1}{2}A_{21}\mathbf{x}_{\text{d}}^*) + \mathbf{x}_{\text{d}}^{*T}\mathbf{x}_{\text{d}}^*.
\end{aligned}$$

□

**Lemma 3.3.5. (*Derivative of Gram Matrix*)** *The determinant of the gram matrix  $Q = A_{21}A_{21}^T$  with respect to the elements of  $A_{21}$  is  $\frac{\partial}{\partial A_{21}} \det(Q) = 2 \det(Q)(Q^{-1}A_{21})$*

*Proof.* For  $A_{21}$ , we note the matrix determinant derivative identity

$$\frac{\partial \det(A_{21}BA_{21})}{\partial A_{21}^T} = \det(A_{21}BA_{21}^T)(B + B^T)A_{21}^T(A_{21}BA_{21}^T)^{-1}.$$

If we set  $B = I$ , this simplifies to

$$\frac{\partial \det(Q)}{\partial A_{21}^T} = 2 \det(Q)A_{21}^T(Q)^{-1}.$$

We note that

$$\frac{\partial \det(Q)}{\partial A_{21}} = \left( \frac{\partial \det(Q)}{\partial A_{21}^T} \right)^T,$$

which ultimately yields

$$\frac{\partial \det(Q)}{\partial A_{21}} = 2 \det(Q) (Q^{-1} A_{21}).$$

□

**Lemma 3.3.6. (Gram Vector Decomposition)** For system matrix  $A_{21}$  with linearly independent rows  $\mathbf{a}_j$ , and symmetric positive definite gram matrix  $Q = A_{21} A_{21}^T = P D P^T$  with eigenvalues  $\lambda_i$  and eigenvectors  $\mathbf{e}_i$ , the total control energy can be represented by  $E(\mathbf{u}) = 12 \left( \frac{\sum_{i=1}^M w_i c_i^2}{\sum_{i=1}^M w_i} \right) \sum_{k=1}^M \frac{1}{\|\mathbf{a}_k\|^2 \sin(\theta_k)^2} + \mathbf{v}_2^T \mathbf{v}_2$ , where  $w_i = \prod_{j \neq i} \lambda_j$ ,  $c_i = \mathbf{e}_i^T \mathbf{v}_1$ , and  $\theta_i$  is the angle formed by  $\mathbf{a}_k$  and the sub-parallelotope formed by the remaining  $\mathbf{a}_{j \neq k}$ .

*Proof.* We recall that the total control energy is given by

$$\begin{aligned} E_{total} &= 12 \mathbf{v}_1^T Q^{-1} \mathbf{v}_1 + \mathbf{v}_2^T \mathbf{v}_2 \\ &= 12 \mathbf{v}_1^T P D^{-1} P^T \mathbf{v}_1 + \mathbf{v}_2^T \mathbf{v}_2 \\ &= 12 (P^T \mathbf{v}_1)^T D^{-1} (P^T \mathbf{v}_1) + \mathbf{v}_2^T \mathbf{v}_2 \\ &= 12 \mathbf{c}^T D^{-1} \mathbf{c} + \mathbf{v}_2^T \mathbf{v}_2 \\ &= 12 \sum_{i=1}^M \frac{c_i^2}{\lambda_i} + \mathbf{v}_2^T \mathbf{v}_2 \end{aligned}$$



We multiply each  $k$  term to find a common denominator to yield

$$\begin{aligned}
E_{total} &= 12 \frac{\sum_{i=1}^M c_i^2 \prod_{j \neq i}^M \lambda_j}{\prod_j^M \lambda_j} + \mathbf{v}_2^T \mathbf{v}_2 \\
&= 12 \frac{\sum_{i=1}^M c_i^2 \prod_{j \neq i}^M \lambda_j}{\det(Q)} + \mathbf{v}_2^T \mathbf{v}_2 \\
&= 12 \left( \frac{\sum_{i=1}^M c_i^2 \prod_{j \neq i}^M \lambda_j}{\sum_{i=1}^M \prod_{j \neq i}^M \lambda_j} \right) \frac{\sum_{k=1}^M \prod_{l \neq k}^M \lambda_l}{\det(Q)} + \mathbf{v}_2^T \mathbf{v}_2.
\end{aligned}$$

We note that the left term is just a weighted average, with weights  $w_i = \prod_{j \neq i}^M \lambda_j$ . We also note that  $\sum_{i=1}^M \prod_{l \neq i}^M \lambda_l$  is the  $M^{th}$  term of the characteristic polynomial of  $Q$ , which is equivalent to  $\sum_{k=1}^M \det(Q_{kk})$ , where  $Q_{kk}$  represents the  $(k, k)$  minor of  $Q$ . Hence, we write

$$E_{total} = 12 \left( \frac{\sum_{i=1}^M c_i^2 w_i}{\sum_{i=1}^M w_i} \right) \frac{\sum_{k=1}^M \det(Q_{kk})}{\det(Q)} + \mathbf{v}_2^T \mathbf{v}_2.$$

We make use of the geometric fact that the determinant of  $Q = A_{21} A_{21}^T$  is equal to the squared volume of the parallelotope formed by the rows of  $A_{21}$ . We also note that minor  $Q_{kk}$  is the gram matrix of  $A_{21}$  after removing  $\mathbf{a}_k$ , represented by  $A_{21}^{k*}$ . Therefore the ratio of the determinants of  $Q_{kk}$  and  $Q$  becomes the squared ratio of parallelotope volumes with and without  $\mathbf{a}_k$ .

$$\frac{\det(Q_{kk})}{\det(Q)} = \frac{\text{vol}(A_{21}^{k*})^2}{\text{vol}(A_{21})^2}.$$

Finally, we realize that the contribution of  $\mathbf{a}_k$  to the parallelotope volume is by a multiple of  $\|\mathbf{a}_k\| \sin(\theta_k)$ , where  $\theta_k$  is the angle formed by  $\mathbf{a}_k$  and the sub-parallelotope, given by  $\text{vol}(A_{21}) = \text{vol}(A_{21}^{k*}) \|\mathbf{a}_k\| \sin(\theta_k)$  to yield

$$E(\mathbf{u}) = 12 \left( \frac{\sum_{i=1}^M w_i c_i^2}{\sum_{i=1}^M w_i} \right) \sum_{k=1}^M \frac{1}{\|\mathbf{a}_k\|^2 \sin(\theta_k)^2} + \mathbf{v}_2^T \mathbf{v}_2.$$

□

**Lemma 3.3.7. (Average Minimum Control Energy)** For final states  $x_i^*$  drawn from independent and identically distributed random variables  $X_i$  with mean  $\mu = 0$  and variance  $c$ , the average minimum control energy to reach all random states is  $\mathbb{E}[E(\mathbf{u})] = c \left( N + 3M + 12 \left( \sum_{k=1}^M \frac{1}{\|\mathbf{a}_k\|^2 \sin(\theta_k)^2} \right) \right)$ .

*Proof.* From Lemma 3.3.4, we have

$$E(\mathbf{u}) = 12(\mathbf{x}_{\text{nd}}^* - \frac{1}{2}A_{21}\mathbf{x}_{\text{d}}^*)^T (A_{21}A_{21}^T)^{-1} (\mathbf{x}_{\text{nd}}^* - \frac{1}{2}A_{21}\mathbf{x}_{\text{d}}^*) + \mathbf{x}_{\text{d}}^{*T} \mathbf{x}_{\text{d}}^*,$$

which is a quadratic form. Setting  $Q = A_{21}A_{21}^T$ , we can reformulate the quadratic form into matrix representation

$$\begin{aligned} E &= \mathbf{x}_{\text{d}}^{*T} \mathbf{x}_{\text{d}}^* + 3\mathbf{x}_{\text{d}}^{*T} A_{21}^T Q^{-1} A_{21} \mathbf{x}_{\text{d}}^* - 6\mathbf{x}_{\text{d}}^{*T} A_{21}^T Q^{-1} \mathbf{x}_{\text{nd}}^* - 6\mathbf{x}_{\text{nd}}^{*T} 6Q^{-1} A_{21} \mathbf{x}_{\text{d}}^* + 12\mathbf{x}_{\text{nd}}^{*T} Q^{-1} \mathbf{x}_{\text{nd}}^* \\ &= \mathbf{x}_{\text{d}}^{*T} (I + 3A_{21}^T Q^{-1} A_{21}) \mathbf{x}_{\text{d}}^* - 6\mathbf{x}_{\text{d}}^{*T} A_{21}^T Q^{-1} \mathbf{x}_{\text{nd}}^* - 6\mathbf{x}_{\text{nd}}^{*T} 6Q^{-1} A_{21} \mathbf{x}_{\text{d}}^* + 12\mathbf{x}_{\text{nd}}^{*T} Q^{-1} \mathbf{x}_{\text{nd}}^* \\ &= \begin{bmatrix} \mathbf{x}_{\text{d}}^{*T} & \mathbf{x}_{\text{nd}}^{*T} \end{bmatrix} \begin{bmatrix} I_N + 3A_{21}^T Q^{-1} A_{21} & -6A_{21}^T Q^{-1} \\ -6Q^{-1} A_{21} & 12Q^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{\text{d}}^* \\ \mathbf{x}_{\text{nd}}^* \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{x}_{\text{d}}^{*T} & \mathbf{x}_{\text{nd}}^{*T} \end{bmatrix} W_R^{-1} \begin{bmatrix} \mathbf{x}_{\text{d}}^* \\ \mathbf{x}_{\text{nd}}^* \end{bmatrix}, \end{aligned}$$

where matrix  $W_R$  is the *Reachability Gramian*. From multivariate statistics, it is known that for symmetric matrix  $W_R^{-1}$ , the expected value of a quadratic form (Mathai and Provost, 1992) is

$$\begin{aligned} \mathbb{E}[E] &= \mathbb{E}[\mathbf{x}^{*T} W_R^{-1} \mathbf{x}^*] \\ &= \text{Tr}(W_R^{-1} \Sigma) + \boldsymbol{\mu}^T W_R^{-1} \boldsymbol{\mu}, \end{aligned}$$

where  $\Sigma$  and  $\boldsymbol{\mu}$  are the covariance matrix and expected values of  $\mathbf{x}^*$ . Because our states  $x_i^* \in X_i$  are drawn from independent and identically distributed random variables  $X_i \sim \text{iid } U(-1, 1)$  with mean 0,  $\boldsymbol{\mu} = \mathbf{0}$ , the covariance  $\text{cov}(X_i, X_j) = 0$  for  $i \neq j$ , and  $\text{cov}(X_i, X_i) =$

$\text{var}(X_i) = c$ , such that  $\Sigma = cI$ . The average minimum control energy simplifies to

$$\begin{aligned}
\mathbb{E}[E] &= \text{Tr}(W_R^{-1}cI) \\
&= c\text{Tr}(W_R^{-1}) \\
&= c\text{Tr}(I_N + 3A_{21}^T Q^{-1} A_{21}) + c\text{Tr}(12Q^{-1}) \\
&= c\text{Tr}(I_N) + c\text{Tr}(3A_{21}^T Q^{-1} A_{21}) + c\text{Tr}(12Q^{-1}) \\
&= cN + 3c\text{Tr}(A_{21}^T Q^{-1} A_{21}) + 12c\text{Tr}(Q^{-1}).
\end{aligned}$$

To further simplify this expression, we use the singular value decomposition of  $A_{21}$  such that  $A_{21} = U\Sigma V^T$ , where  $U$  is an  $M \times M$  unitary matrix,  $V$  is an  $N \times N$  unitary matrix, and  $\Sigma$  is an  $M \times N$  rectangular diagonal matrix with elements  $\Sigma_{ii} = \sqrt{\lambda_i}$ , where  $\lambda_i$  are the eigenvalues of  $A_{21}A_{21}^T$ . Then our average control energy simplifies to

$$\begin{aligned}
\mathbb{E}[E] &= cN + 3c\text{Tr}(A_{21}^T Q^{-1} A_{21}) + 12c\text{Tr}(Q^{-1}) \\
&= cN + 3c\text{Tr}(V\Sigma^T U^T (U\Sigma V^T V\Sigma^T U^T)^{-1} U\Sigma V^T) + 12c\text{Tr}(Q^{-1}) \\
&= cN + 3c\text{Tr}(V\Sigma^T U^T (U\Sigma\Sigma^T U^T)^{-1} U\Sigma V^T) + 12c\text{Tr}(Q^{-1}) \\
&= cN + 3c\text{Tr}(V\Sigma^T U^T U(\Sigma\Sigma^T)^{-1} U^T U\Sigma V^T) + 12c\text{Tr}(Q^{-1}) \\
&= cN + 3c\text{Tr}(V\Sigma^T D^{-1} \Sigma V^T) + 12c\text{Tr}(Q^{-1}),
\end{aligned}$$

where  $D$  is a diagonal matrix of eigenvalues of  $A_{21}A_{21}^T$ . We realize that if  $D_{ii} = \lambda_i$ , then  $D_{ii}^{-1} = \frac{1}{\lambda_i}$ , such that

$$\begin{aligned}
\Sigma^T D^{-1} \Sigma &= \begin{bmatrix} \sqrt{D} \\ 0 \end{bmatrix} \begin{bmatrix} D^{-1} \end{bmatrix} \begin{bmatrix} \sqrt{D} & 0 \end{bmatrix} \\
&= \begin{bmatrix} I_M & 0 \\ 0 & 0 \end{bmatrix},
\end{aligned}$$

where  $I_M$  is the  $M \times M$  identity matrix. Finally, we can reduce the average control energy

to

$$\begin{aligned}
\mathbb{E}[E] &= cN + 3cTr \left( V \begin{bmatrix} I_M & 0 \\ 0 & 0 \end{bmatrix} V^T \right) + 12cTr(Q^{-1}) \\
&= cN + 3cTr \left( \sum_{i=1}^M \mathbf{v}_i \mathbf{v}_i^T \right) + 12cTr(Q^{-1}) \\
&= cN + 3c \sum_{i=1}^M Tr(\mathbf{v}_i \mathbf{v}_i^T) + 12cTr(Q^{-1}) \\
&= cN + 3c \sum_{i=1}^M \mathbf{v}_i^T \mathbf{v}_i + 12cTr(Q^{-1}) \\
&= cN + 3c \sum_{i=1}^M 1 + 12cTr(Q^{-1}) \\
&= cN + 3cM + 12c \sum_{i=1}^M \frac{1}{\lambda_i}.
\end{aligned}$$

Finally, we recall from Lemma 3.3.6 that

$$\begin{aligned}
\sum_{k=1}^M \frac{1}{\lambda_k} &= \frac{\sum_{k=1}^M \prod_{l \neq k} \lambda_l}{\det(Q)} \\
&= \frac{\sum_{k=1}^M \det(Q_{kk})}{\det(Q)} \\
&= \sum_{k=1}^M \frac{1}{\|\mathbf{a}_k\|^2 \sin(\theta_k)^2},
\end{aligned}$$

which is the topology-based term, to yield

$$\mathbb{E}[E(\mathbf{u})] = c \left( N + 3M + 12 \left( \sum_{k=1}^M \frac{1}{\|\mathbf{a}_k\|^2 \sin(\theta_k)^2} \right) \right).$$

We note that for our particular uniform distribution between  $-a$  and  $a$ , the variance is

$$c = 4a^2 \frac{1}{12} = \frac{1}{3},$$

such that the average minimum control energy becomes

$$\mathbb{E}[E(\mathbf{u})] = \frac{1}{3}N + M + 4 \left( \sum_{k=1}^M \frac{1}{\|\mathbf{a}_k\|^2 \sin(\theta_k)^2} \right).$$

□

### 3.4. Mathematical Framework: Second-Order Energy Approximation

Here, we extend our approximation of the minimum control energy to begin incorporating non-driver to non-driver connections as given by matrix  $A_{22}$ . We begin with the same control dynamics given in Eq. 3.2. In the first-order energy approximation, we only considered connections from driver to non-driver nodes. Here, we extend this approximation to include one additional layer of non-driver to non-driver nodes.

**Assumption 3.** *The initial state of the network satisfies  $\mathbf{x}_d(0) = 0$  and  $\mathbf{x}_{nd}(0) = 0$ .* □

**Assumption 4.** *The network  $\mathcal{G}$  contains only edges from the drivers to the non-drivers, and one layer of connections from non-drivers to the non-drivers. That is,  $A_{11} = 0$ ,  $A_{12} = 0$ , and  $A_{22}^k = 0$  for  $k > 1$ .* □

From assumptions 3 and 4, we notice an interesting phenomena with respect to the powers of  $A$ , such that

$$\begin{aligned} A^1 &= \begin{bmatrix} 0 & 0 \\ A_{21} & A_{22} \end{bmatrix}, \\ A^2 &= \begin{bmatrix} 0 & 0 \\ A_{22}A_{21} & A_{22}A_{22} \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 0 \\ A_{22}A_{21} & 0 \end{bmatrix}, \\ A^3 &= \begin{bmatrix} 0 & 0 \\ A_{22}A_{22}A_{21} & A_{22}A_{22}A_{22} \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \end{aligned}$$

such that all powers of  $A$  greater than 2 yield the zero matrix. We also readily observe that

$$\begin{aligned}
\mathbf{x}_d(t) &= \int_0^t \mathbf{u}(\tau) d\tau, \text{ and} \\
\mathbf{x}_{nd}(t) &= A_{21} \int_0^t \mathbf{x}_d(\tau) d\tau + A_{22} \int_0^t \mathbf{x}_{nd}(\tau) d\tau \\
&= A_{21} \int_0^t \mathbf{x}_d(\tau) d\tau + A_{22}A_{21} \int_0^t \int_0^\tau \mathbf{x}_d(\tau_1) d\tau_1 d\tau \\
&= A_{21} \int_0^t \int_0^\tau \mathbf{u}(\tau_1) d\tau_1 d\tau + A_{22}A_{21} \int_0^t \int_0^\tau \int_0^{\tau_1} \mathbf{u}(\tau_2) d\tau_2 d\tau_1 d\tau.
\end{aligned} \tag{3.5}$$

**Lemma 3.4.1. (*Controllability, Second-Order*)** *The network (2.1) is controllable if and only if  $\text{Rank} \left( \begin{bmatrix} A_{21} & A_{22}A_{21} \end{bmatrix} \right) = M$ . Furthermore, the set of controllable states is  $\text{Im} \left( \begin{bmatrix} I_N & 0 & 0 \\ 0 & A_{21} & A_{22}A_{21} \end{bmatrix} \right)$ .*

*Proof.* Notice that the controllability matrix of (2.1) is

$$C = \begin{bmatrix} B & AB & A^2B \end{bmatrix} = \begin{bmatrix} I_N & 0 & 0 \\ 0 & A_{21} & A_{22}A_{21} \end{bmatrix},$$

and recall that a state is controllable if and only if it belongs to the range space of the controllability matrix.  $\square$

**Lemma 3.4.2. (*Total Minimum Control Energy, Second-Order*)** *The total control energy is  $12\mathbf{v}_1^T(KK^T)^{-1}\mathbf{v}_1 + \mathbf{v}_2^T\mathbf{v}_2$ , where  $\mathbf{v}_1 = \mathbf{x}_{nd}^* - \left(\frac{1}{2}A_{21} + \frac{1}{6}A_{22}A_{21}\right)\mathbf{x}_d^*$ ,  $\mathbf{v}_2 = \mathbf{x}_d^*$ , and  $K = \begin{bmatrix} A_{21} + \frac{1}{2}A_{22}A_{21} & \frac{1}{2\sqrt{15}}A_{22}A_{21} \end{bmatrix}$ . Further, the system is controllable iff  $\text{Rank}(K) = M$ .*

*Proof.* Recall that the minimum control energy can be given by the *Reachability Gramian*

$$W_R(0, T) = \int_0^T e^{At} B B^T e^{A^T t} dt,$$

where the energy required to reach final state  $\mathbf{x}(T) = \mathbf{x}^*$  takes the form

$$E(\mathbf{u}) = \mathbf{x}^{*T} W_R^{-1} \mathbf{x}^*.$$

We write  $W_R$  at time  $T = 1$  using the node reordering given by Eq. 3.2, and expand the matrix exponentials in conjunction with assumptions 3 and 4 to yield

$$\begin{aligned} W_R(0, 1) &= \int_0^1 e^{At} B B^T e^{A^T t} dt \\ &= \begin{bmatrix} I_N & \frac{1}{2} A_{21}^T + \frac{1}{6} A_{21}^T A_{22}^T \\ \frac{1}{2} A_{21} + \frac{1}{6} A_{22} A_{21} & \frac{1}{3} Q + \frac{1}{8} (Q A_{22}^T + A_{22} Q) + \frac{1}{20} A_{22} Q A_{22}^T \end{bmatrix}, \end{aligned}$$

from which we can derive the Schur complement

$$\begin{aligned} S_D &= \frac{1}{3} Q + \frac{1}{8} (Q A_{22}^T + A_{22} Q) + \frac{1}{20} A_{22} Q A_{22}^T - (\frac{1}{2} A_{21} + \frac{1}{6} A_{22} A_{21}) (\frac{1}{2} A_{21}^T + \frac{1}{6} A_{21}^T A_{22}^T) \\ &= \frac{1}{3} Q + \frac{1}{8} (Q A_{22}^T + A_{22} Q) + \frac{1}{20} A_{22} Q A_{22}^T - \frac{1}{4} Q - \frac{1}{12} (Q A_{22}^T + A_{22} Q) - \frac{1}{36} A_{22} Q A_{22}^T \\ &= \frac{1}{12} Q + \frac{1}{24} Q A_{22}^T + \frac{1}{24} A_{22} Q + \frac{1}{45} A_{22} Q A_{22}^T \\ &= \frac{1}{12} (Q + \frac{1}{2} Q A_{22}^T + \frac{1}{2} A_{22} Q + \frac{4}{15} A_{22} Q A_{22}^T), \end{aligned}$$

yielding the inverse of the Reachability Gramian

$$W_R^{-1} = \begin{bmatrix} I + (\frac{1}{2} A_{21} + \frac{1}{6} A_{22} A_{21})^T S_D^{-1} (\frac{1}{2} A_{21} + \frac{1}{6} A_{22} A_{21}) & -(\frac{1}{2} A_{21} + \frac{1}{6} A_{22} A_{21})^T S_D^{-1} \\ -S_D^{-1} (\frac{1}{2} A_{21} + \frac{1}{6} A_{22} A_{21}) & S_D^{-1} \end{bmatrix},$$

with minimum control energy

$$\begin{aligned} E &= \begin{bmatrix} \mathbf{x}_d^{*T} & \mathbf{x}_{nd}^{*T} \end{bmatrix} W_R^{-1} \begin{bmatrix} \mathbf{x}_d^* \\ \mathbf{x}_{nd}^* \end{bmatrix} \\ &= (\mathbf{x}_{nd}^* - (\frac{1}{2} A_{21} + \frac{1}{6} A_{22} A_{21}) \mathbf{x}_d^*)^T S_D^{-1} (\mathbf{x}_{nd}^* - (\frac{1}{2} A_{21} + \frac{1}{6} A_{22} A_{21}) \mathbf{x}_d^*) + \mathbf{x}_d^{*T} \mathbf{x}_d^*. \end{aligned}$$

We can decompose the Schur complement into a Gram matrix such that  $S_D = \frac{1}{12} K K^T$ ,

where

$$K = \begin{bmatrix} A_{21} + \frac{1}{2}A_{22}A_{21} & \frac{1}{2\sqrt{15}}A_{22}A_{21} \end{bmatrix},$$

which gives us the first result.

Next, we observe that  $S_D$  is a Gram matrix, and is therefore positive semi-definite. Hence, the control energy to reach any final state  $\mathbf{x}_{\text{nd}}^*, \mathbf{x}_{\text{d}}^*$  is the sum of the bilinear product of  $\mathbf{v}_1$  with  $S_D^{-1}$ , and  $\mathbf{x}_{\text{d}}^{*T} \mathbf{x}_{\text{d}}^*$ . If  $\text{Rank}(K) = M$ , then  $\text{Rank}(S_D) = \text{Rank}(S_D^{-1}) = M$ , and all states can be reached with finite energy. However, if  $\text{Rank}(K) = k < M$ , then  $\text{Rank}(S_D) = \text{Rank}(S_D^{-1}) = k < M$ , and there exists a  $M - k$  dimensional subspace of unreachable states.  $\square$

**Lemma 3.4.3. (Gram Vector Decomposition, Second-Order)** *For the following matrix  $K = \begin{bmatrix} A_{21} + \frac{1}{2}A_{22}A_{21} & \frac{1}{2\sqrt{15}}A_{22}A_{21} \end{bmatrix}$  with linearly independent rows  $\mathbf{a}_j$ , and symmetric positive definite gram matrix  $Q = KK^T = PDP^T$  with eigenvalues  $\lambda_i$  and eigenvectors  $\mathbf{e}_i$ , the total control energy can be represented by  $E(\mathbf{u}) = 12 \left( \frac{\sum_{i=1}^M w_i c_i^2}{\sum_{i=1}^M w_i} \right) \sum_{k=1}^M \frac{1}{\|\mathbf{a}_k\|^2 \sin(\theta_k)^2} + \mathbf{v}_2^T \mathbf{v}_2$ , where  $w_i = \prod_{j \neq i}^M \lambda_j$ ,  $c_i = \mathbf{e}_i^T \mathbf{v}_1$ , and  $\theta_i$  is the angle formed by  $\mathbf{a}_k$  and the sub-parallelotope formed by the remaining  $\mathbf{a}_{j \neq k}$ .*

*Proof.* We recall that the total control energy is given by

$$\begin{aligned} E_{\text{total}} &= 12\mathbf{v}_1^T Q^{-1} \mathbf{v}_1 + \mathbf{v}_2^T \mathbf{v}_2 \\ &= 12\mathbf{v}_1^T P D^{-1} P^T \mathbf{v}_1 + \mathbf{v}_2^T \mathbf{v}_2 \\ &= 12(P^T \mathbf{v}_1)^T D^{-1} (P^T \mathbf{v}_1) + \mathbf{v}_2^T \mathbf{v}_2 \\ &= 12\mathbf{c}^T D^{-1} \mathbf{c} + \mathbf{v}_2^T \mathbf{v}_2 \\ &= 12 \sum_{i=1}^M \frac{c_i^2}{\lambda_i} + \mathbf{v}_2^T \mathbf{v}_2 \end{aligned}$$



We multiply each  $k$  term to find a common denominator to yield

$$\begin{aligned}
E_{total} &= 12 \frac{\sum_{i=1}^M c_i^2 \prod_{j \neq i}^M \lambda_j}{\prod_j^M \lambda_j} + \mathbf{v}_2^T \mathbf{v}_2 \\
&= 12 \frac{\sum_{i=1}^M c_i^2 \prod_{j \neq i}^M \lambda_j}{\det(Q)} + \mathbf{v}_2^T \mathbf{v}_2 \\
&= 12 \left( \frac{\sum_{i=1}^M c_i^2 \prod_{j \neq i}^M \lambda_j}{\sum_{i=1}^M \prod_{j \neq i}^M \lambda_j} \right) \frac{\sum_{k=1}^M \prod_{l \neq k}^M \lambda_l}{\det(Q)} + \mathbf{v}_2^T \mathbf{v}_2.
\end{aligned}$$

We note that the left term is just a weighted average, with weights  $w_i = \prod_{j \neq i}^M \lambda_j$ . We also note that  $\sum_{i=1}^M \prod_{l \neq i}^M \lambda_l$  is the  $M^{th}$  term of the characteristic polynomial of  $Q$ , which is equivalent to  $\sum_{k=1}^M \det(Q_{kk})$ , where  $Q_{kk}$  represents the  $(k, k)$  minor of  $Q$ . Hence, we write

$$E_{total} = 12 \left( \frac{\sum_{i=1}^M c_i^2 w_i}{\sum_{i=1}^M w_i} \right) \frac{\sum_{k=1}^M \det(Q_{kk})}{\det(Q)} + \mathbf{v}_2^T \mathbf{v}_2.$$

We make use of the geometric fact that the determinant of  $Q = KK^T$  is equal to the squared volume of the parallelotope formed by the rows of  $K$ . We also note that minor  $Q_{kk}$  is the gram matrix of  $K$  after removing  $\mathbf{a}_k$ , represented by  $K^{k*}$ . Therefore the ratio of the determinants of  $Q_{kk}$  and  $Q$  becomes the squared ratio of parallelotope volumes with and without  $\mathbf{a}_k$ .

$$\frac{\det(Q_{kk})}{\det(Q)} = \frac{\text{vol}(K^{k*})^2}{\text{vol}(K)^2}.$$

Finally, we realize that the contribution of  $\mathbf{a}_k$  to the parallelotope volume is by a multiple of  $\|\mathbf{a}_k\| \sin(\theta_k)$ , where  $\theta_k$  is the angle formed by  $\mathbf{a}_k$  and the sub-parallelotope, given by  $\text{vol}(A_{21}) = \text{vol}(A_{21}^{k*}) \|\mathbf{a}_k\| \sin(\theta_k)$  to yield

$$E(\mathbf{u}) = 12 \left( \frac{\sum_{i=1}^M w_i c_i^2}{\sum_{i=1}^M w_i} \right) \sum_{k=1}^M \frac{1}{\|\mathbf{a}_k\|^2 \sin(\theta_k)^2} + \mathbf{v}_2^T \mathbf{v}_2.$$

□

### 3.5. Validity of the First-Order Approximation

Until now, we have derived several useful closed-form expressions from the first-order minimum energy approximation by making the simplifying assumptions 1, 2. We explore how well this energy approximation holds when we relax the topological assumption 2. As a generalized, analytic closed-form energy solution for non-simplified networks is typically intractable or not informative, we compare the first-order energy approximation to a numerical computation of the unsimplified control energy. From linear control theory, we know that for an LTI system  $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$  obeying the dynamics in (2.1), we can define the *Reachability Gramian*:

$$W_R(0, T) = \int_0^T e^{\mathbf{A}t} \mathbf{B} \mathbf{B}^T e^{\mathbf{A}^T t} dt.$$

The minimum control energy takes the form

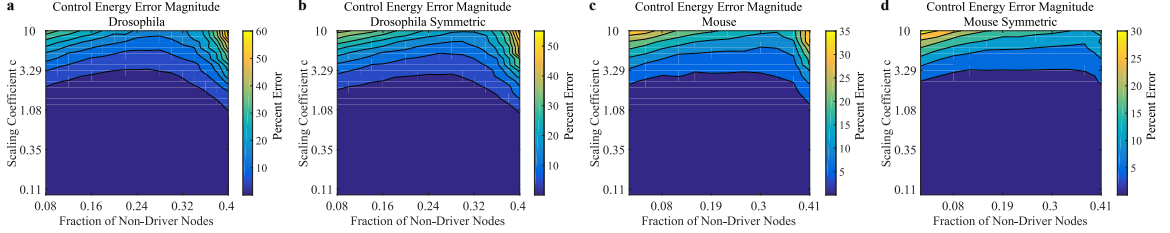
$$\begin{aligned} E_{NS}(\mathbf{u}) &= \sum_{i=1}^N \int_0^T u_i(t)^2 dt \\ &= (\mathbf{W}_R^{-1} \mathbf{x}^*)^T \mathbf{W}_R (\mathbf{W}_R^{-1} \mathbf{x}^*) \\ &= \mathbf{x}^{*T} \mathbf{W}_R^{-1} \mathbf{x}^*, \end{aligned}$$

from which we calculate the percent error between the minimum energy of the simplified *versus* unsimplified network.

### 3.6. Comparison of Results for Directed & Undirected Networks

While invasive tract-tracing methods allow for the resolution of directionality in the *Drosophila* and mouse connectomes, the non-invasive diffusion spectrum imaging methods used for the human connectomes are unable to resolve directionality. To ensure that the main results between the *Drosophila*, mouse, and human networks are not due to directed *versus* undirected matrices, we compare the directed and undirected mouse and *Drosophila* connectomes to

the undirected human connectome. For the undirected mouse and Drosophila connectomes, we will model each directed edge as two edges in opposite directions. Specifically, for directed network  $A$ , we model undirected network  $\hat{A} = A + A^T$ , and normalize  $\hat{A}$  by the magnitude of its largest eigenvalue.



**Figure 7: Similar Accuracy of First-Order Energy Approximation.** (a) Percent error contour plots of the total control energy for the simplified *versus* non-simplified networks as a function of the fraction of non-driver nodes and the matrix scale (given by the magnitude of the largest eigenvalue  $c = \|\lambda_{\max}\|$ ), in the directed Drosophila, (b) undirected Drosophila, (c) directed mouse, and (d) undirected mouse networks.

First, we compare the accuracy of the first-order energy approximation between the directed and undirected versions of the Drosophila and mouse connectomes (Fig. 7a–d) using the same methodology in result A. As can be seen, there is little difference in the accuracy of the first-order approximation between the directed and undirected versions of either the Drosophila or mouse connectomes. We note that the intuition provided in result B does not use any of the three networks, and will therefore not be included in this analysis.

Next, we reproduce result C comparing the energetic performance and connectivity for the energetically most and least favorable non-driver selections between the directed and undirected Drosophila and mouse connectomes (Fig. 8a–h). As can be seen, there is little difference in the energetic performance of the energetically most and least favorable non-driver selections between the directed and undirected versions of the Drosophila and mouse connectomes (Fig. 8a–d). There also remains a statistically significant difference in the non-driver weight-vector’s magnitude and angle product ( $\|\mathbf{a}_k\| \sin(\theta_k)$ ) between the energetically most and least favorable non-driver selections of the directed and undirected versions of the Drosophila and mouse connectomes (Fig. 8e–h).

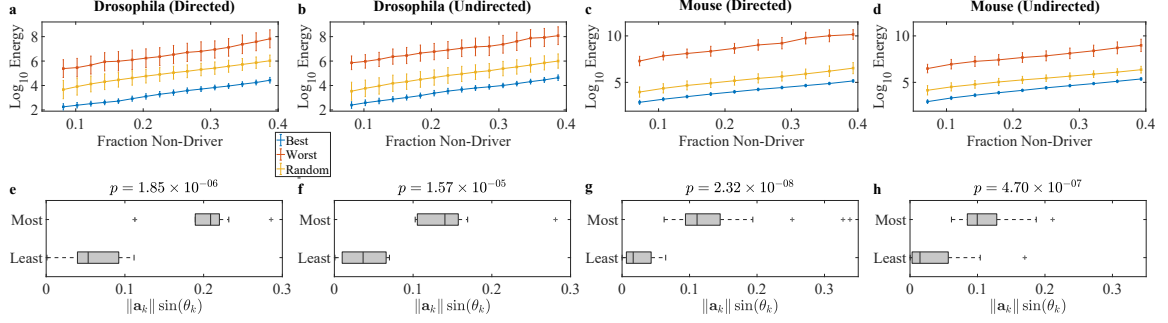


Figure 8: **Similar Distributions of Control Energy & Topology.** (a) Means and standard deviations of the base-10 log of the total control energies across various non-driver fractions to reach 2000 random final states in the energetically most favorable, least favorable, and random non-driver selections for the directed Drosophila, (b) undirected Drosophila, (c) directed mouse, and (d) undirected mouse networks. (e) Boxplots of the distribution of each non-driver weight-vector's magnitude and angle product  $\|a_k\| \sin(\theta_k)$  between the energetically most and least favorable networks for the directed Drosophila, (f) undirected Drosophila, (g) directed mouse, and (h) undirected mouse connectomes. Above each boxplot is the  $p$ -value of the 2-sample  $t$ -test between the most and least favorable networks.

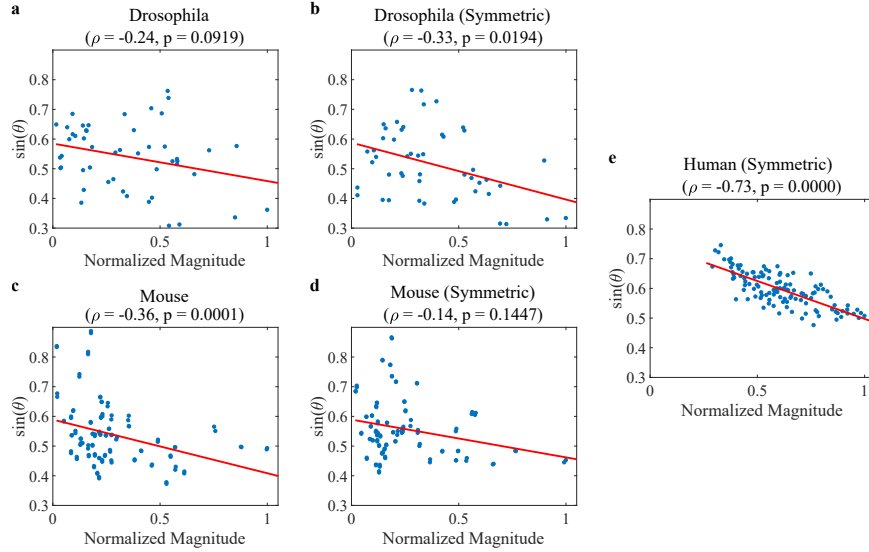


Figure 9: **Similar Energetically Favorable Organization.** (a) Plots of the average magnitude *versus*  $\sin(\theta)$  for each brain region across 10,000 random non-driver selections at a non-driver fraction of 0.2 for the directed Drosophila, (b) undirected Drosophila, (c) directed mouse, (d) undirected mouse, and (e) undirected human connectomes. Above each plot is the Spearman rank correlation coefficient ( $\rho$ ) and  $p$ -value.

Next, we reproduce result D for the directed and undirected versions of the Drosophila and mouse connectomes, and compare them to the undirected human connectome. We

see that there is little qualitative difference in the relationship between average non-driver magnitude and angle for the directed *versus* undirected versions of the Drosophila (Fig. 9a,b) and mouse (Fig. 9c,d) connectomes. Quantitatively, we notice neither the directed nor undirected Drosophila and mouse connectome is as energetically favorably organized (as given by the Spearman rank correlation coefficient) as the human (Fig. 9e). This relation holds true across a wide range of non-driver fractions (Fig. 10a,b).

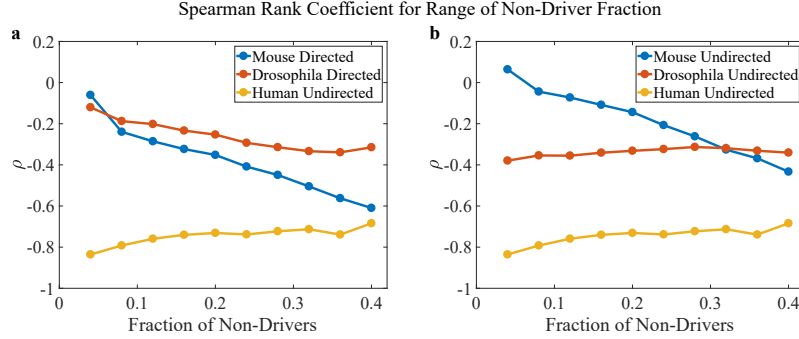


Figure 10: **The Order of the Correlation is Maintained between Species.** (a) Plot of Spearman rank correlation coefficient  $\rho$  for 10,000 random non-driver selections across a range of non-driver fractions for the directed Drosophila, directed mouse, and undirected human connectomes. (b) Spearman rank coefficients for the undirected mouse, undirected Drosophila, and undirected human connectomes.

Finally, we reproduce result E (at a non-driver fraction of 0.2) to compare the percent reduction in energy after an energetically favorable edge deletion between directed and undirected versions of the mouse and Drosophila connectomes. As can be seen, there is qualitatively little difference in the distribution of percent change in energy between the directed and undirected versions of the mouse and Drosophila connectomes.

### 3.7. Differences in Network Density do not Drive Differences in Results Between Networks

Here, we elucidate the role of network density on our results to ensure that any inherent differences in average connectivity due to differing data acquisition techniques are not responsible for any of our results. To assess the role of network density, we study 5 random 100 node networks of densities 0.2, 0.4, 0.6, 0.8, and 1.0, whose elements are randomly

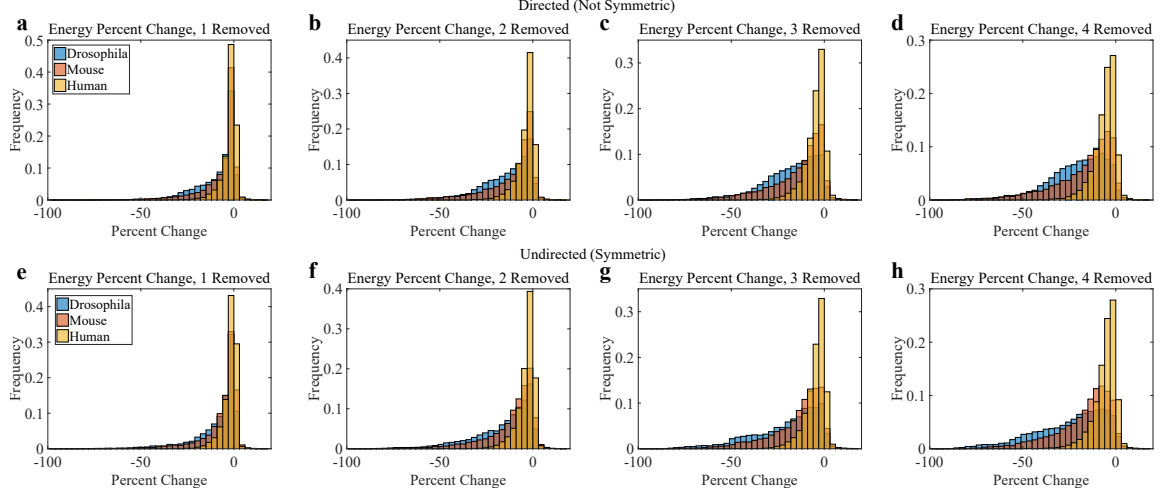


Figure 11: **Similar Changes in Control Energy for Edge Deletions.** (a) Distribution of percent change in control energy over 10,000 random selections of non-driver nodes to reach 10,000 random final states at non-driver fraction of 0.2 for the directed Drosophila and mouse networks after 1 edge removed, (b) 2 edges removed, (c) 3 edges removed, and (d) 4 edges removed. (e) Same distributions of percent change in control energy in the undirected Drosophila and mouse networks after 1 edge removed, (f) 2 edges removed, (g) 3 edges removed, and (h) 4 edges removed.

selected from a uniform distribution between 0 and 1, and then normalize each network by dividing by the magnitude of its largest eigenvalue. In the connectomes used in this paper, the Drosophila has density 0.81, the mouse 0.52, and the human 0.26.

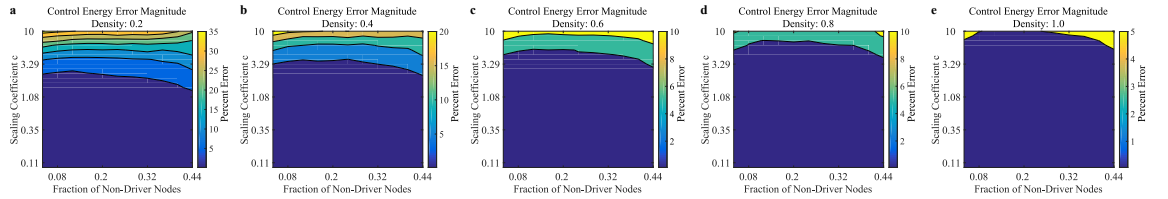
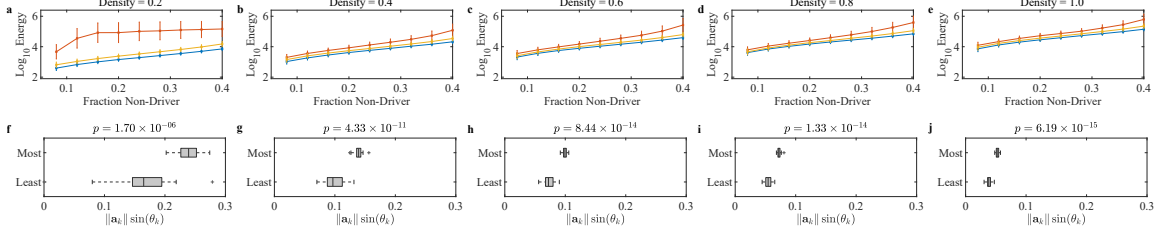


Figure 12: **Percent Error in Energy Improves for Increasing Densities.** (a) Percent error contour plots of the total control energy for the simplified *versus* non-simplified networks as a function of the fraction of non-driver nodes and the matrix scale (given by the magnitude of the largest eigenvalue  $c = \|\lambda_{\max}\|$ ) for network densities of 0.2, (b) 0.4, (c) 0.6, (d) 0.8, and (e) 1.0.

First, we study the role of network density on the goodness of the first-order approximation by reproducing the contour plots in result A for these 5 random networks (Fig. 12a–e). As can be seen, the first-order approximation seems to be more accurate with increasing

network density. However, in all cases, the approximation works well ( $< 5\%$ ) within our parameters of interest (matrix scale  $c = \|\lambda_{\max}\| = 1$ , non-driver fraction  $d \leq 0.4$ ).



**Figure 13: Non-Driver Selection Based on the Topology-Dependent Term Yields Significantly Different Energetic Performance across Densities.** (a) Means and standard deviations of the base-10 log of the total control energies across various non-driver fractions to reach 2000 random final states in the energetically most favorable, least favorable, and random selections of non-drivers for network density 0.2, (b) 0.4, (c) 0.6, (d) 0.8, and (e) 1.0. (f) Boxplots of the distribution of each non-driver weight-vector’s topology term  $\|\mathbf{a}_k\| \sin(\theta_k)$  between the energetically most and least favorable non-driver selections at network density 0.2, (g) 0.4, (h) 0.6, (i) 0.8, and (j) 1.0, with corresponding  $p$ -values from a 2-sample  $t$ -test.

Next, we reproduce result C for these 5 random networks, and see that across all densities, the energetic performance is best for the energetically most favorable non-driver selections, and worst for the energetically least favorable non-driver selections (Fig. 13a–e). We also see that the non-driver weight-vector’s topology term  $\|\mathbf{a}_k\| \sin(\theta_k)$  is statistically significantly smaller for the energetically least favorable network *versus* that of the energetically most favorable network across all densities (Fig. 13f–j).

Next we reproduce result D at a non-driver fraction of 0.2, and find that while increasing network density does significantly change the distribution of magnitudes and angles of the non-driver weight-vectors, the relationship between network density and the monotonically decreasing relationship between magnitudes and angles (as given by the Spearman rank correlation coefficient) is not clear (Fig. 14a–e). Hence, we generated at 100 instantiations of random networks at each density, and computed the Spearman rank correlation coefficient between the average magnitudes and angles for each instantiation (for 500 total rank coefficients, 100 coefficients for each density). We show boxplots of the distribution of Spearman rank as a function of network density (Fig. 14f), and see that the Spearman rank is less

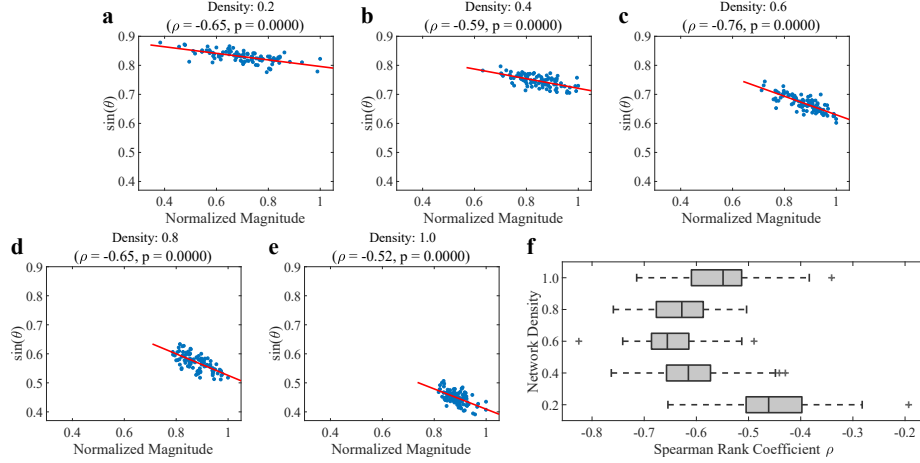


Figure 14: **Non-Monotonic Relationship between Spearman Rank Correlation Coefficient & Network Density.** (a) Plots of the average magnitude *versus*  $\sin(\theta)$  for each node across 10,000 random non-driver selections at a non-driver fraction of 0.2 at random network densities of 0.2, (b) 0.4, (c) 0.6, (d) 0.8, and (e) 1.0. (f) Distribution of Spearman rank correlation coefficients across 100 instantiations of random networks for each density between 0.2–1.0.

negative for very low (0.2) and very high (1.0) densities. According to this relationship, the human connectome (density = 0.26) should have the least negative Spearman rank, but in reality has the most negative Spearman rank. Hence we see that network density does not drive the negative relationship between magnitude and angle seen in our connectomes.

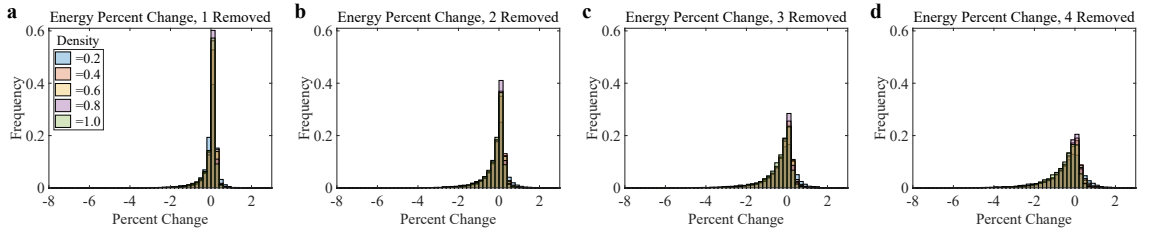


Figure 15: **Similar Distributions for Changes in Control Energy for Energetically Favorable Edge Deletions across Varying Densities.** (a) Distribution of percent change in control energy over 10,000 random selections of non-driver nodes to reach 10,000 random final states at non-driver fraction of 0.2 at each network density between 0.2–1.0 for 1 edge deleted, (b) 2 edges deleted, (c) 3 edges deleted, and (d) 4 edges deleted.

Finally, we reproduce result E across these random networks by computing the percent change in energy after deleting the edge that maximally increases the Gram determinant, and find that there is negligible difference in percent energy change across the 5 densities



(Fig. 15a–d).

In summary, we show that there is nothing inherent about the densities of the connectomes used in the paper that would yield the relationships in energetically favorable connectivity we observe between the *Drosophila*, mouse, and human connectomes.

### 3.8. Dealing with Driver & Non-Driver Allocations that Yield Unreachable States

In our simplified network topologies involving only connections from driver to non-driver nodes, there may be a situation where a non-driver node is not connected to any driver nodes, such that the system is not fully reachable. Here, we study the reachability of our networked systems, and elaborate on how we handled unreachable cases.

From classical linear control theory, we guarantee that a network is *reachable* such that we can control the states from 0 to any point in the state space if the reachability Gramian  $W_R$  has full rank. That is, for  $A, B \in \mathbb{R}^{n \times n}$ ,  $\mathbf{x}, \mathbf{u} \in \mathbb{R}^{n \times 1}$  which obeys dynamics

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u},$$

we have guaranteed reachability when

$$\text{rank}(W_R) = \text{rank} \left( \int_0^T e^{At} B B^T e^{A^T t} dt \right) = n.$$

In our simplified networks involving  $N$  driver nodes,  $M$  non-driver nodes, and only driver  $\rightarrow$  non-driver connections, we have guaranteed reachability when

$$\text{rank}(A_{21} A_{21}^T) = M.$$

In our analysis for result A, we select 1000 random permutations of drivers and non-drivers for a set of 30 matrix scaling coefficients and around 12 non-driver fractions, for a total of

$1000 \times 30 \times 12 = 360,000$  trials per species. Among these trials, we encountered around 10 cases where the simplified network was not fully reachable, and 0 cases where the full network was not fully reachable. Due to the very few instances of not-fully-reachable selections, these specific selections were not included in the computations of percent error.

In our analysis for result B, the mathematical and geometric representations require that  $\text{rank}(A_{21}A_{21}^T) = M$ . If  $A_{21}$  does not have full rank, then the representation as shown in result B does not apply, which is a limitation of the analysis.

In our analysis for result C, we ensure that the energetically most and least favorable selections of drivers and non-drivers is fully reachable through the requirement that nodes are selected as non-drivers only if  $\text{rank}(A_{21}A_{21}^T) = M$ . Further elaboration of this selection process can be found in the online methods under *Selection of Energetically Most and Least Favorable Non-Drivers*. In the selection of random networks for result C, we again encountered a negligible number of driver and non-driver selections which were not fully reachable, and omitted them from the analysis.

In our analysis for result D, we compute 10,000 random non-driver selections per non-driver fraction, where only a negligible fraction ( $< 0.1\%$ ) of selections were not fully reachable. Hence, we omitted these selections from computing the average magnitudes and angles of our connectomes.

In our analysis for result E, we have the same situation as the previous results, where very few ( $< 0.1\%$ ) driver and non-driver designations were not fully reachable. Hence, we omitted these selections from consideration to guarantee that the percent changes in control energy are computed only on reachable networks.

### 3.9. Matrix Scaling Retains System Properties

Throughout the results, we use matrices that are normalized by the magnitude of their largest eigenvalue. Here, we demonstrate that this normalization does not change the

global stability or instability of our system, nor does it change the qualitative dynamics of individual eigenmodes. Because we study continuous linear time-invariant systems that obey

$$\dot{\mathbf{x}} = A\mathbf{x},$$

our systems are unstable if any eigenvalue of  $A$  has positive real component. We note that the number of eigenvalues with positive *versus* negative real components are 18 to 31 in the *Drosophila*, 38 to 74 in the mouse, and 49 to 67 in the human, such that all of our connectomes are unstable. We also know that if  $\lambda_i$  and  $\mathbf{e}_i$  are the  $i$ -th eigenvalue and eigenvector of  $A$ , then  $c\lambda_i$  and  $\mathbf{e}_i$  are the  $i$ -th eigenvalue and eigenvector of  $cA$ . Hence, as long as we scale our matrix by a positive constant  $c > 0$ , the sign of the real component of our eigenvalues does not change, and the stability of each eigenmode does not change.

Next, we note that if the  $i$ -th eigenvalue of  $A$  has an imaginary component such that the time-evolution of that particular eigenmode is oscillatory, then scaling the system by a positive real constant  $c > 0$  does not remove the imaginary component, preserving oscillations. Conversely, if the  $i$ -th eigenvalue of  $A$  has no imaginary component, then scaling the system by a positive real constant cannot add oscillations. Hence, scaling the system by a positive real constant also preserves the qualitative behavior of individual eigenmodes.

### 3.10. Retaining Goodness of Approximation for Scaled Matrices

Throughout the results, we normalize each of our system matrices by the largest eigenvalue to normalize the most unstable dynamic response, as well as to remain at a scale for which the first-order energy approximation is close to the non-simplified control energy. Here, we provide a compromise between matrix scale and duration of the control window to attain scale invariance in goodness of approximation. Specifically, given the linear control dynamics outlined in Eq. 2.1, we can utilize classic results in linear control theory to define the *Reachability Gramian*  $W_R$  such that the minimum energy required to drive our system

to final state  $\mathbf{x}^*$  is

$$\begin{aligned} E_{\text{NS}}(A, T) &= \mathbf{x}^{*T} W_R(A, T)^{-1} \mathbf{x}^* \\ &= \mathbf{x}^{*T} \left( \int_0^T e^{At} B B^T e^{A^T t} dt \right)^{-1} \mathbf{x}^*. \end{aligned}$$

Suppose we wish to consider scaled system matrix  $cA$ , where  $c$  is a positive real number. If we also scale the control time window to  $\frac{T}{c}$ , then the minimum energy to bring system  $cA$  to final state  $\mathbf{x}^*$  in time  $\frac{T}{c}$  becomes

$$\begin{aligned} E_{\text{NS}}\left(cA, \frac{T}{c}\right) &= \mathbf{x}^{*T} \left( \int_0^{\frac{T}{c}} e^{cAt} B B^T e^{cA^T t} dt \right)^{-1} \mathbf{x}^* \\ &= \mathbf{x}^{*T} \left( \int_0^{\frac{T}{c}} e^{A(ct)} B B^T e^{A^T(ct)} dt \right)^{-1} \mathbf{x}^* \\ &= \mathbf{x}^{*T} \left( \int_0^T e^{A\tau} B B^T e^{A^T \tau} \frac{1}{c} d\tau \right)^{-1} \mathbf{x}^* \\ &= \mathbf{x}^{*T} c \left( \int_0^T e^{A\tau} B B^T e^{A^T \tau} d\tau \right)^{-1} \mathbf{x}^* \\ &= c \mathbf{x}^{*T} W_R(A, T)^{-1} \mathbf{x}^* \\ &= c E_{\text{NS}}(A, T). \end{aligned}$$

Hence we see that scaling matrix  $A$  by  $c$  and driving the system to final states in  $\frac{T}{c}$  time costs  $c$  times more energy. This linear scaling by  $c$  is useful when we recognize that the first-order energy approximation considered in the paper is the same as the energy detailed here (and thereby obeys the same linear scaling by  $c$ ), where  $\hat{A} = \begin{bmatrix} 0 & 0 \\ A_{21} & 0 \end{bmatrix}$ , such that

$$E_S\left(c\hat{A}, \frac{T}{c}\right) = c E_S(\hat{A}, T).$$

Let us call the unscaled percent error in control energy between the simplified *versus* non-simplified networks

$$e(A, T) = \frac{E_S(A, T) - E_{NS}(A, T)}{E_{NS}(A, T)}.$$

Then, the percent error after scaling both systems to  $cA$  and  $c\hat{A}$  becomes

$$\begin{aligned} e\left(cA, \frac{T}{c}\right) &= \frac{E_S\left(cA, \frac{T}{c}\right) - E_{NS}\left(cA, \frac{T}{c}\right)}{E_{NS}\left(cA, \frac{T}{c}\right)} \\ &= \frac{cE_S(A, T) - cE_{NS}(A, T)}{cE_{NS}(A, T)} \\ &= e(A, T), \end{aligned}$$

such that the percent error of the control energy is scale invariant. Hence, if we wish to consider larger matrix  $cA$  where  $c > 1$ , but wish to retain the same goodness of approximation, we simply have to reduce the control time window to  $\frac{T}{c}$ .

### 3.11. Analysis & Data Acquisition Retains Significant Biological Consistency Between Networks

Given the large decreases in energy after removal of the energetically most favorable edge deletion(s), we consider the importance of connectome data integrity on our results. Due to the simplicity of our dynamical model and control paradigm, we desire to know if our methods and results are able to capture a significant degree of biologically meaningful information despite imperfect connectome data.

In particular, we study result E regarding the selection of the energetically most favorable edge deletion across the 10 human connectomes. Specifically, if our methods are unable to extract biologically significant structural features, then we expect that there will be little relationship between the deleted edges across our 10 human connectomes. However, if our analyses are able to extract these features such that important biological similarities are

retained, then we expect the energetically most favorable edge deletion to be shared between human connectomes.

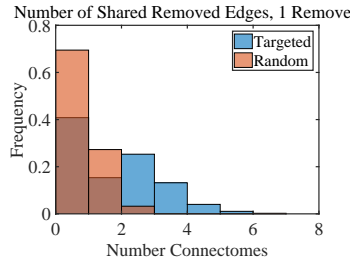


Figure 16: **Higher Number of Human Connectomes Share Energetically Favorable Edges for Deletion than Randomly Selected Edges.** Distribution of the number of connectomes that share the same edge between targeted selection of the energetically most favorable edge deletion, and random edge selection, across 10,000 random selections of drivers and non-drivers at a non-driver fraction of 0.2.

Using this motivation, we selected the same 10,000 random permutations of drivers and non-drivers (at non-driver fraction 0.2) across our 10 human connectomes. For each selection, we found the energetically most favorable driver  $\rightarrow$  non-driver edge deletion across all 10 connectomes, and computed the maximum number of connectomes (1–10) that shared the same deleted edge. For the same 10,000 random permutations, we selected a random driver  $\rightarrow$  non-driver edge for each connectome, and computed the maximum number of connectomes that shared a randomly selected edge. A histogram of the maximum number of connectomes sharing either the energetically most favorable edge deletion or the random edge deletion is shown across the 10,000 driver and non-driver selections (Fig. 16).

As can be seen, there is a significantly higher number of connectomes that share the energetically most favorable edge deletion than the randomly selected edge. Hence, we show that our analysis can detect some biologically consistent feature in these 10 similar human connectomes better than chance.

### 3.12. Value of Mathematically Formalizing the Intuitive Concept of Differential Connectivity

One of the main goals of these results was to produce an intuitive analytic and geometric representation of complex network structure with respect to control energy. This representation had an equally intuitive implication for improving controllability: differentially connected networks require less energy to bring the system to distinct outputs. One reason why it is valuable to quantify this notion of “differential connectivity” is because the networks we study are often immense in size.

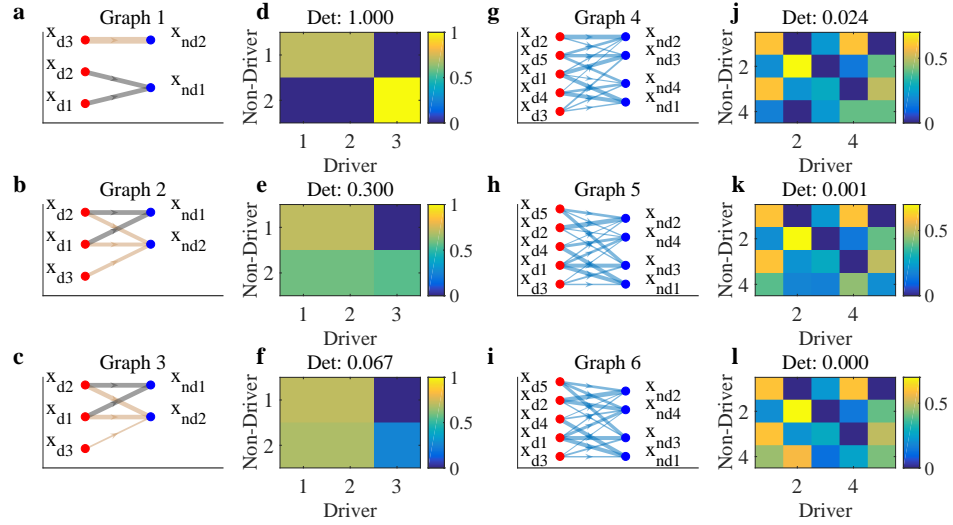


Figure 17: **Differential Connectivity Difficult to Identify with Increasing Network Size.** (a) Graph representation of a 5 node network with 3 drivers (red) and 2 non-drivers (blue) that is very differentially connected, (b) somewhat differentially connected, and (c) not differentially connected. (d) Image of connectivity matrix  $A_{21}$  with corresponding determinant of  $A_{21}A_{21}^T$  for Graph 1, (e) Graph 2, and (f) Graph 3. (g) Graph representation of a 9 node network with 5 drivers (red) and 4 non-drivers (blue) that is very differentially connected, (h) somewhat differentially connected, and (i) not differentially connected. (j) Image of connectivity matrix  $A_{21}$  with corresponding determinant of  $A_{21}A_{21}^T$  for Graph 4, (k) Graph 5, and (l) Graph 6.

Using a simple 5 node toy network (3 drivers, 2 non-drivers), we can tell fairly easily by inspection of the graph representation (Fig. 17a–c) or the corresponding matrix representation  $A_{21}$  (Fig. 17d–f) how differentially connected the non-driver nodes are. However,

even when expanding to a 9 node toy network (5 drivers, 4 non-drivers), it is very difficult to tell by inspection of the graph representation (Fig. 17g–i) or corresponding matrix representation (Fig. 17j–l) which of graphs 4–6 are more differentially connected. Hence, the mathematical formalization allows us to generalize the simple and intuitive concept of differential connectivity to understand large and complex network topologies.

Another reason why it is advantageous to mathematically formalize this concept of differential connectivity is because this formalization allows us to generalize this concept to more complex network topologies. Specifically, we derive all of the same results and intuitions of the paper, but also taking into account non-driver to non-driver connections encoded in matrix  $A_{22}$  beginning in section 3.13. In particular, this framework of understanding network connectivity as differentially connected vectors extends to include higher order network connectivity (e.g. self-loops, cycles), such that we can begin fully understanding and modifying complex network topology using a simple and intuitive framework. We present a motivating example of the utility of this framework in modifying a simple 3 node network (1 driver, 2 non-drivers) involving higher-order network topology to improve control (Fig. 19a–k).

### 3.13. Validity of the Second-Order Approximation

Using the derivation of the second-order approximation of the control energy, we perform the same analysis in result A across a larger fraction of non-driver nodes in the *Drosophila*, mouse, and human connectomes (Fig. 18).

As can be seen, the second-order energy approximation remains very accurate for scaling coefficient  $c \leq 1$  to 60% non-driver fraction. For the duration of the supplementary results, we will use scaling coefficient  $c = \|\lambda_{\max}\| = 1$  and keep the non-driver fraction  $d \leq 0.6$ .



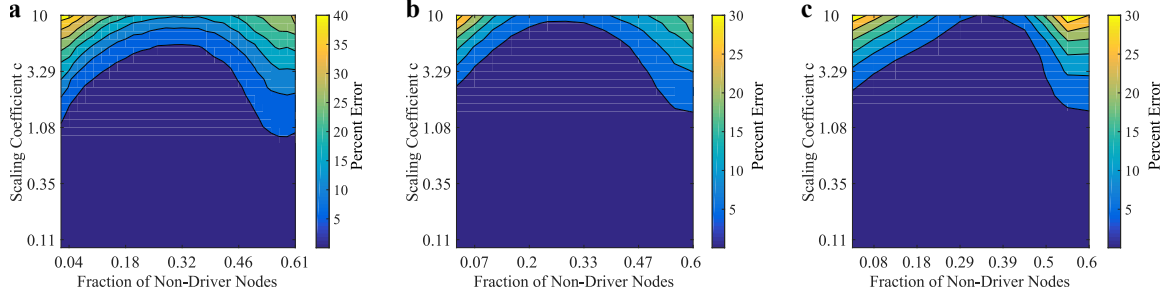


Figure 18: **The Second-Order Energy Approximation Offers a Reasonable Prediction for the Full Network’s Control Energy for Higher Non-Driver Fractions.** (a) Percent error contour plots of the total control energy for simplified *versus* non-simplified networks as a function of the fraction of non-driver nodes and matrix scale given by  $c = \|\lambda_{\max}\|$ . For each combination of parameters, the median error magnitude to drive the networks from initial states  $\mathbf{x}_d = 0$ ,  $\mathbf{x}_{nd} = 0$  to 1000 random final states  $\mathbf{x}_{nd}^* \in (-1, 1)^M$ ,  $\mathbf{x}_d^* \in (-1, 1)^N$  along 1000 corresponding random selections of non-drivers is shown. Each contour represents a 5% interval for the (a) Drosophila, (b) mouse, and (c) human connectome.

### 3.14. Determinant of the Second-Order Connectivity Matrix Scales the Control Energy

In the main text, we demonstrated that the rows of  $A_{21}$  represented the vector of connections  $\mathbf{a}_k$  from all of the driver nodes to the  $k$ -th non-driver node. In the second-order formulation, we take the rows  $\mathbf{a}_k$  of matrix  $K$  to be the geometric vectors that form a parallelotope whose squared volume is still equal to the Gram determinant  $\det(KK^T)$ . However, as the rows of  $K$  are not as simple as in the previous result, we leave the full and formal intuition for the relationship between the rows of  $K$  and network connectivity for future work. Here, we provide the general case of a simple three node toy network example with one driver and two non-drivers (Fig. 19a). We study three specific instances of this three node network (Graphs 1, 2, and 3), where the edges contributing to the vector formed by the first row of  $K$  are highlighted in gray (Fig. 19b–d), the edges contributing to the vector formed by the second row of  $K$  are highlighted in tan (Fig. 19e–g), and the geometric parallelogram (and corresponding Gram determinant) formed by these two vectors are shown for each graph (Fig. 19h–j).

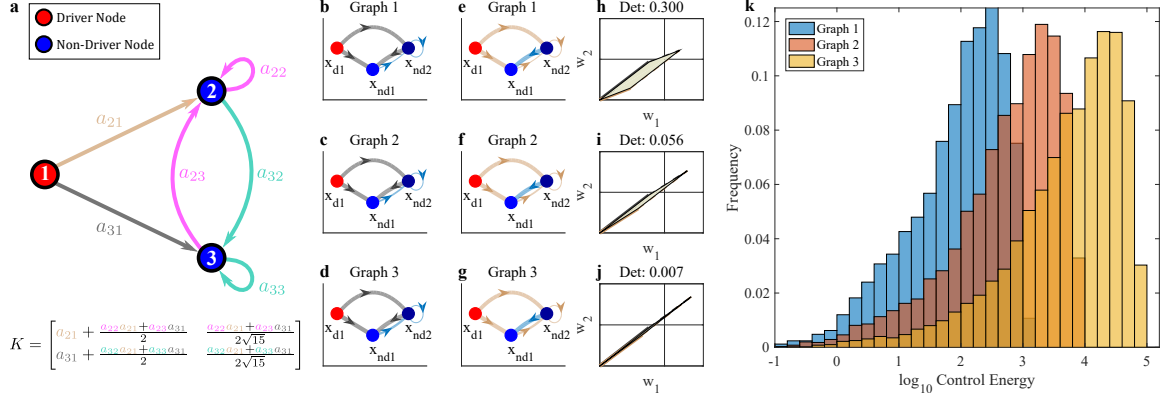


Figure 19: **Geometric Example of Simplified, Second-Order Networks with Corresponding Control Energies.** (a) Graph representation of a simple three node network, with one driver (node 1, red) and two non-drivers (nodes 2 and 3, blue). The terms of the corresponding matrix  $K$  are written with respect to the contributing edges in the network. (b) Graph representation of a three node network (one driver, two non-drivers) with the edges contributing to the first row of  $K$  in gray for dissimilarly distributed weights, (c) somewhat similarly distributed weights, and (d) very similarly distributed weights. (e) Graph representation of the same three node network with edges contributing to the second row of  $K$  in tan for dissimilarly distributed weights, (f) somewhat similarly distributed weights, and (g) very similarly distributed weights. (h) Geometric representation of the paralleloptope formed by the two rows of  $K$  representing the paths from the driver node to non-driver nodes 1 and 2, with the volume shaded in beige and the value of  $\det(KK^T)$  above each plot for dissimilarly distributed weights, (i) somewhat similarly distributed weights, and (j) very similarly distributed weights. (k) Distribution of the base-10 log control energy to bring the three corresponding graphs into 10,000 random final states  $\mathbf{x}_{\text{nd}}^* \in (-1, 1)^M$   $\mathbf{x}_{\text{d}}^* \in (-1, 1)^N$ .

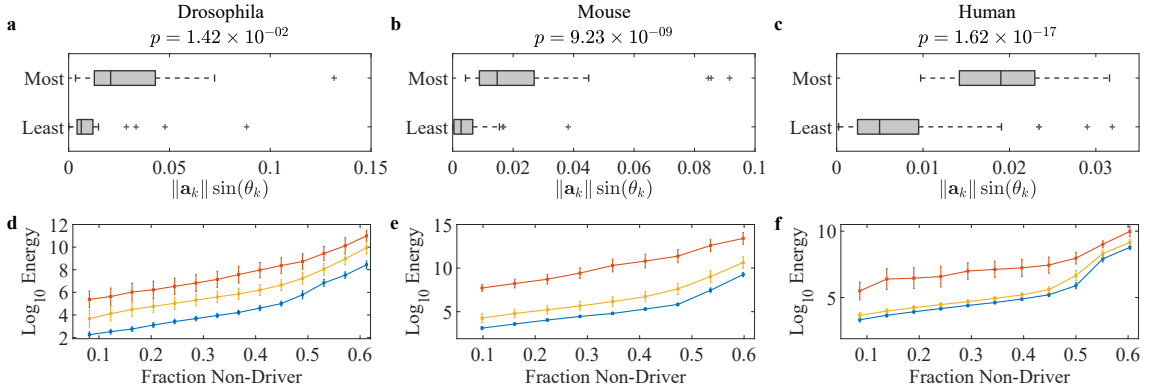
We note that the only difference between these graphs is that we slightly increase the strength of connection from non-driver  $x_{\text{nd}1}$  to non-driver  $x_{\text{nd}2}$ , which corresponds to a decrease in the parallelogram area. This slight change yields an order of magnitude increase in the control energy distribution to bring our system to 10,000 random final states between Graphs 1 and 2, and another order of magnitude energy increase between Graphs 2 and 3 (Fig. 19k).

### 3.15. Most & Least Energetically Favorable Driver-Non-Driver Sets in Brain Connectomes Using Second-Order Approximation

Next, we examine the selections of drivers and non-drivers to generate the energetically most and least favorable networks using the second-order approximation. We show in Lemma 3.4.3 that the decomposition of the control energy retains the same form as in the main result C

$$E(\mathbf{u}) = 12 \left( \frac{\sum_{i=1}^M w_i c_i^2}{\sum_{i=1}^M w_i} \right) \sum_{k=1}^M \frac{1}{\|\mathbf{a}_k\|^2 \sin(\theta_k)^2} + \mathbf{v}_2^T \mathbf{v}_2,$$

where vector  $\mathbf{a}_k$  is the  $k$ -th row of matrix  $K = \begin{bmatrix} A_{21} + \frac{1}{2} A_{22} A_{21} & \frac{1}{2\sqrt{15}} A_{22} A_{21} \end{bmatrix}$ .



**Figure 20: Topological Characteristics & Energetic Performance of Networks using the Second-Order Approximation.** (a) Boxplots of the magnitude and angle product  $\|\mathbf{a}_k\| \sin(\theta_k)$  for each second-order non-driver weight-vector  $\mathbf{a}_k$  (that is, the  $k$ -th row of  $K$ ) for a non-driver fraction of 0.5 for the Drosophila, (b) mouse, and (c) human. (d) Means and standard deviations of the base-10 log of the total control energies to reach 2000 random final states along the energetically most favorable, least favorable, and random networks for the Drosophila, (e) mouse, and (f) human.

As can be seen in Fig. 20a–c, we retain a statistically significant difference in the topology term  $\|\mathbf{a}_k\| \sin(\theta_k)$  for the Drosophila, mouse, and human at a non-driver fraction of 0.5. We also see that there remains a significant difference between the control energy required to control the most and least energetically favorable networks across a wide range of non-drivers for the Drosophila, mouse, and human (Fig. 20d–f).

### 3.16. Brain Networks of Increasingly Complex Species Have More Energetically Favorable Second-Order Organization of Connectivity Features

Using the second-order approximation, we examine the average magnitude *versus*  $\sin(\theta)$  for each node in the Drosophila, mouse, and human networks across 10,000 random non-driver selections at a non-driver fraction of 0.5. The vectors  $\mathbf{a}_k$  of the second-order approximation come from the rows of matrix  $K = \begin{bmatrix} A_{21} + \frac{1}{2}A_{22}A_{21} & \frac{1}{2\sqrt{15}}A_{22}A_{21} \end{bmatrix}$  (Fig. 21a–c). As can be seen, the monotonically decreasing relationship between species (as measured by the Spearman rank correlation coefficient  $\rho$ ) is least negative for the Drosophila, second least negative for the mouse, and most negative for the human.

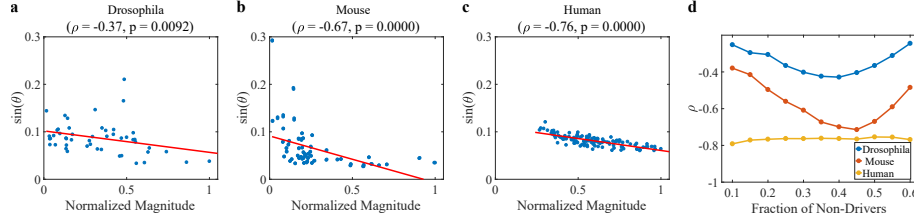


Figure 21: **Energetically Favorable Organization of Second-Order Topological Features in Networks.** (a) Average  $\sin(\theta_k)$  *versus* normalized magnitude  $\|\mathbf{a}_k\|$  (where  $\mathbf{a}_k$  represents the  $k$ -th row of matrix  $K$ ) for each brain region across 10,000 random non-driver selections for a non-driver fraction of 0.5, along with the best-fit line (red) and corresponding Spearman correlation coefficient in the Drosophila, (b) mouse, and (c) human.

### 3.17. Network Manipulation to Facilitate Control Using the Second-Order Approximation

Here, we use the second-order approximation to find edge deletions in our networks that maximally increase the Gram determinant  $\det(KK^T)$ , and we compare the energies required to control our network before and after 1–4 edge deletions (Fig. 22a–d).

As can be seen, the ordering of decreasing percent change in energy after edge deletion (Drosophila  $\rightarrow$  mouse  $\rightarrow$  human) persists across a wide range of non-driver fractions and number of edge deletions. However, there is an anomalous change for the human connectome

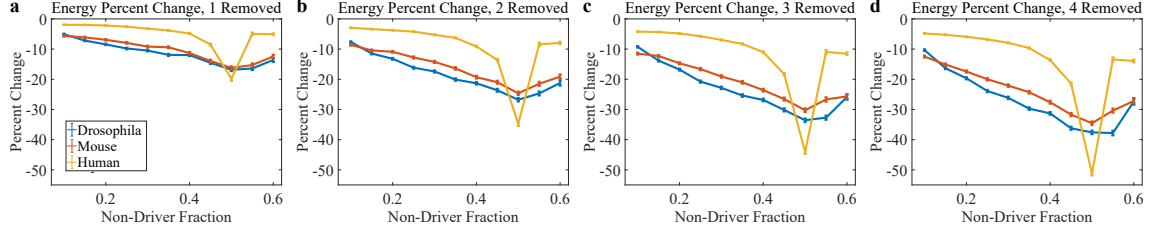


Figure 22: **Modifying the Connectomes to Decrease Minimum Energy Using both Driver to Non-Driver, as well as Non-Driver to Non-Driver Connections.** (a) Means and standard errors of the percent change in energy after deleting edges that maximally increase the Gram determinant  $\det(KK^T)$  to reach 2,000 random final states after 1 edge deletion, (b) 2 edge deletions, (c) 3 edge deletions, and (d) 4 edge deletions across a range of non-driver fractions. Standard errors were computed as  $SE = \frac{s}{\sqrt{n}}$ , where  $s$  is the sample standard deviation over the 2,000 tasks, and  $n = 2,000$ .

at a non-driver fraction of 0.5, where the percent change in energy becomes dramatically more negative. We also see some other interesting characteristics around 0.5. For non-driver fractions below 0.5, the percent change in energy tends to become more negative with increasing non-driver fraction. However, above 0.5, the percent change in energy becomes less negative with increasing non-driver fraction. We hypothesize there is some regime switch between a primary dependence on first-order connections below 0.5 non-driver fraction, and a primary dependence on second-order connections above 0.5 non-driver fraction.

### 3.18. Maximizing the Determinant Through Node Selection

In result B of the main text, we represented the connectivity of driver to non-driver connections  $A_{21}$  as vectors, and posed the problem of reducing control energy as an exercise in increasing  $\det(A_{21}A_{21}^T)$ . Similarly in result E, we simulated edge deletions that increase this determinant to reduce the energy to control the Drosophila, mouse, and human connectomes. Here, we explore this relationship between control energy and determinant more thoroughly.

For each species, we select 4000 random permutations of driver and non-driver nodes at a non-driver fraction of 0.2. For each permutation, we extract driver to non-driver connections

$A_{21}$ , and compute  $\det(A_{21}A_{21}^T)$  and the average control energy to bring the non-simplified network to 5,000 random final states  $\mathbf{x}_d^* \in (-1, 1)^N$ ,  $\mathbf{x}_{nd}^* \in (-1, 1)^M$  (Fig. 23).

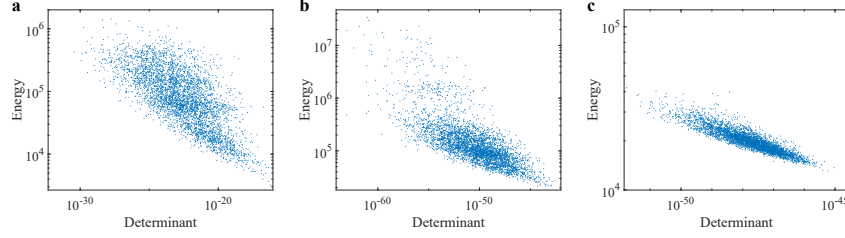


Figure 23: **Decreasing Average Energy as a Function of Increasing Determinant in Brain Networks.** (a) Average control energy required to drive the simplified system of 20% non-drivers involving only driver  $\rightarrow$  non-driver connections to 5,000 random final states  $\mathbf{x}_d^* \in (-1, 1)^N$ ,  $\mathbf{x}_{nd}^* \in (-1, 1)^M$  as a function of  $\det(A_{21}A_{21}^T)$  for 4,000 possible permutations in the Drosophila, (b) mouse, and (c) human connectomes. For clarification, each plot has 4,000 points, where each point represents the determinant and average energy of one of 4,000 configurations of drivers and non-drivers.

As can be seen, there is a strong negative relationship where increasing the Gram determinant decreases the average control energy. It is not guaranteed that the selection of drivers and non-drivers that yields the largest determinant will necessarily cost the lowest energy to control. However, in general there is a very strong negative trend between the determinant and average control energy.

### 3.19. Maximizing the Determinant Minimizes Average Control Energy for $A_{21}$ with Fixed Frobenius Norm

In result B, we demonstrate that increasing the determinant of a toy system of 3 drivers and 2 non-drivers decreases the average energy required to control that system. Here, we prove that for all  $M \times N$  matrices  $A_{21}$  where  $N > M$  with fixed Frobenius norm, the matrix  $A_{21}$  which maximizes  $\det(A_{21}A_{21}^T)$  actually does minimize the average control energy. To begin, we consider matrix  $A_{21} \in \mathbb{R}^{M \times N}$  where  $N > M$  bounded by the Frobenius norm

$$\|A_{21}\|_F^2 = \sum_{i=1}^M \sum_{j=1}^N a_{ij}^2 = \sum_{i=1}^M \|\mathbf{a}_i\|^2 = \text{Tr}(A_{21}A_{21}^T) = \sum_{i=1}^M \lambda_i = C.$$

This norm provides the double benefit of explicitly bounding the elements of  $A_{21}$  as well as the eigenvalues of the Gram matrix of interest. To maximize the determinant  $\det(A_{21}A_{21}^T) = \lambda_1\lambda_2\cdots\lambda_M$ , we seek

$$\begin{aligned} \max_{\lambda_1, \lambda_2, \dots, \lambda_M} \det(A_{21}A_{21}^T) &= \lambda_1\lambda_2\cdots\lambda_M \\ \text{subject to: } \sum_{i=1}^M \lambda_i &= C, \end{aligned}$$

which has known solution  $\lambda_1 = \lambda_2 = \cdots = \lambda_M = \frac{C}{M}$ . We note that  $A_{21}$  can always be selected such that the eigenvalues of  $A_{21}A_{21}^T$  are  $\lambda_1 = \lambda_2 = \cdots = \lambda_M = \frac{C}{M}$  by making the rows of  $A_{21}$  orthogonal with squared magnitude  $\|\mathbf{a}_k\|^2 = \frac{C}{M}$ . Then  $A_{21}A_{21}^T$  simply becomes a diagonal matrix with entries  $\mathbf{a}_k\mathbf{a}_k^T = \|\mathbf{a}_k\|^2 = \frac{C}{M}$ , and the eigenvalues of a diagonal matrix are just the diagonal entries such that  $\lambda_i = \|\mathbf{a}_i\|^2 = \frac{C}{M}$ .

Now we turn our attention to minimizing the average control energy. From Lemma 3.3.7, we have that for final states  $x_i^*$  drawn from independent and identically distributed random variables  $X_i$  with mean  $\mu = 0$  and variance  $c$ , the average minimum control energy can be written as

$$\begin{aligned} \mathbb{E}[E(\mathbf{u})] &= c \left( N + 3M + 12 \left( \sum_{k=1}^M \frac{1}{\|\mathbf{a}_k\|^2 \sin(\theta_k)^2} \right) \right) \\ &= c \left( N + 3M + 12 \sum_{k=1}^M \frac{1}{\lambda_k} \right). \end{aligned}$$

To minimize the average control energy, we must find

$$\begin{aligned} \min_{\lambda_1, \lambda_2, \dots, \lambda_M} Tr(Q^{-1}) &= \sum_{i=1}^M \frac{1}{\lambda_i} \\ \text{subject to: } \sum_{i=1}^M \lambda_i &= C, \end{aligned}$$

from which, using the method of Lagrange multipliers, we can formulate the Lagrange

function

$$\begin{aligned}\mathcal{L}(\lambda_1, \lambda_2, \dots, \lambda_M, \lambda) &= \sum_{i=1}^M \frac{1}{\lambda_i} - \lambda \left( \sum_{i=1}^M \lambda_i - C \right) \\ &= \lambda_1^{-1} + \lambda_2^{-1} + \dots + \lambda_M^{-1} - \lambda(\lambda_1 + \lambda_2 + \dots + \lambda_M - C),\end{aligned}$$

whose gradient

$$\nabla_{\lambda_1, \lambda_2, \dots, \lambda_M, \lambda} \mathcal{L} = \left\{ -\lambda_1^{-2} - \lambda, \quad -\lambda_2^{-2} - \lambda, \quad \dots, \quad -\lambda_M^{-2} - \lambda, \quad \lambda_1 + \lambda_2 + \dots + \lambda_M - C \right\},$$

we set to zero

$$-\lambda_1^{-2} - \lambda = 0, \quad -\lambda_2^{-2} - \lambda = 0, \quad \dots, \quad -\lambda_M^{-2} - \lambda = 0, \quad \lambda_1 + \lambda_2 + \dots + \lambda_M - C = 0.$$

Solving for  $\lambda$ , we get

$$\lambda_1 = \lambda_2 = \dots = \lambda_M = \sqrt{-\frac{1}{\lambda}},$$

such that

$$\begin{aligned}\lambda_1 + \lambda_2 + \dots + \lambda_M - C &= M \sqrt{-\frac{1}{\lambda}} - C = 0 \\ \lambda &= -\frac{M^2}{C^2}.\end{aligned}$$

Plugging the Lagrange multiplier back in to solve for  $\lambda_i$ , we get solution  $\lambda_1 = \lambda_2 = \dots = \lambda_M = \frac{C}{M}$ . Hence, we prove that for  $A_{21} \in \mathbb{R}^{M \times N}$  with fixed Frobenius norm, maximizing  $\det(A_{21} A_{21}^T)$  yields the same solution as minimizing the average control energy.



## CHAPTER 4 : Teaching Recurrent Neural Networks to Infer Global Structure

### 4.1. Motivation

Computers analyze massive quantities of data with speed and precision. At both the hardware and software levels, this performance depends on fixed and precisely engineered protocols for representing and executing basic operations on binary data (von Neumann, 1993; Alglave et al., 2008). In contrast, neurobiological systems are characterized by flexibility and adaptability. At the biophysical level, neurons undergo dynamic changes in their composition and patterns of connectivity (Zhang et al., 2011; Faulkner et al., 2008; Dunn and Wong, 2012; Craik and Bialystok, 2006). At the cognitive level, they abstract spatiotemporally complex sensory information to recognize objects, localize spatial position, and even control new virtual limbs through experience (Tacchetti et al., 2018; Moser et al., 2008; Ifft et al., 2013). Hence, neural systems appear to work on fundamentally different computing principles that are learned, rather than engineered.

To uncover these principles, artificial neural networks have been used to study the representation and manipulation of information. While feed-forward networks can classify input data (Sainath et al., 2015), biological organisms contain recurrent connections that are necessary to continuously sustain short-term memory of internal representations (Jarrell et al., 2012), allowing for more complex functions such as tracking time, distance, and emotional context (Lee and Tashev, 2015; Wang et al., 2018; Weber et al., 2017; Burak and Fiete, 2009; Yoon et al., 2013). Further, recurrent neural systems actually manipulate internal representations as observed in meta-learning and adaptive networks (Kumar et al., 2020; Schweighofer and Doya, 2003; Santiago, 2004; Feldkamp et al., 1997) to simulate the outcome of dynamic processes such as kinematic motion and navigation (Hegarty, 2004; Kubricht et al., 2017; Pfeiffer and Foster, 2013), and to decide between different actions (Gold and Shadlen, 2007). How do recurrent neural systems learn to represent and manipulate complex information?

One promising line of work involves representing static memories as patterns of neural ac-

tivity, or *attractors*, to which a network evolves over time (Strogatz, 1994). These attractors can exist in isolation (e.g. an image of a face) or as a continuum (e.g. smooth translations of a face) using Hopfield or continuous attractor neural networks (CANNs), respectively (Yang et al., 2017; Wu et al., 2016). Other studies engineer neural connectivity as in the Neural Engineering Framework or the differentiable neural computer to encode, modify, and decode internal representations or to solve complex puzzles (Eliasmith and Anderson, 2003; Bekolay et al., 2014; Graves et al., 2016). For understanding neurobiological systems, these memory networks are limited by requiring specifically engineered patterns of connectivity, or cannot manipulate time-varying memories necessary to plan and produce speech and music (Carroll, 2004; Fee and Scharff, 2010; Donnay et al., 2014). Hence, we seek a single neural system that learns to both represent and manipulate temporally complex information by perceiving and replicating examples.

In this work, we use the *reservoir computing framework* (Qiao et al., 2017) to obtain such a system (the reservoir), where the complex information is a chaotic attractor that is not static, but evolves in a deterministic yet unpredictable manner through time (Lorenz, 1963). Prior work has demonstrated an RNN’s ability to represent and switch between isolated attractors by imitating examples (Jaeger, 2010; Sussillo and Abbott, 2009), and to interpolate and extrapolate underlying generative dynamical processes in time-series prediction tasks using fixed-weight neural networks (FWNN) (Feldkamp et al., 1997; Tyukin et al., 2008; Santiago, 2004; Klos et al., 2020). Here, we demonstrate that reservoirs can further learn to interpolate and extrapolate translations, linear transformations, and even bifurcations on their representations of chaotic attractor manifolds simply by imitating examples. Further, reservoirs can infer the bifurcation structure of dynamical normal forms and period doubling routes to chaos, as well as accurately extrapolate non-dynamical, kinematic trajectories. Finally, we put forth a mechanism of how these computations are learned, providing insights into the set of possible computations, and offering principles by which to design networks.

## 4.2. Mathematical Framework

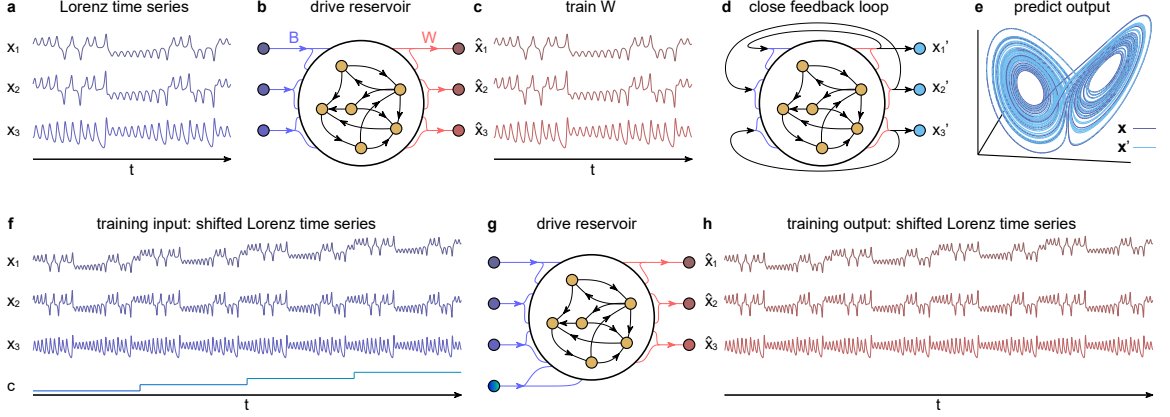


Figure 24: **Representing Chaotic Attractors with Reservoirs.** (a) Time series of a chaotic Lorenz attractor that (b) drives the recurrent neural network reservoir. (c) Weighted sums of the reservoir states are trained to reproduce the original time series. (d) By using these weighted sums of reservoir states to drive the reservoir instead of the inputs, (e) the reservoir autonomously evolves along a trajectory that projects to a Lorenz-shaped chaotic manifold. (f) Time series of shifted copies of the Lorenz input along  $x_1$  and control inputs that (g) drive the reservoir. (h) Weighted sums of the reservoir states are used to mimic the shifted Lorenz inputs.

Neural systems represent and manipulate periodic stimuli through example, such as baby songbirds modifying their song to imitate adult songbirds (Fee and Scharff, 2010). However, they also perform more advanced and original manipulations on aperiodic stimuli with higher-order structure, such as musicians improvising on jazz melodies (Donnay et al., 2014). To model such complex stimuli, we use chaotic attractors that evolve deterministically yet unpredictably along a global structure: a fractional-dimensional manifold. Specifically, we consider the Lorenz attractor defined as

$$\begin{aligned}\dot{x}_1 &= \sigma(x_2 - x_1) \\ \dot{x}_2 &= x_1(\rho - x_3) - x_2 \\ \dot{x}_3 &= x_1x_2 - \beta x_3,\end{aligned}\tag{4.1}$$

and use the parameters  $\sigma = 10, \beta = 8/3, \rho = 28$  from the original study (Lorenz, 1963).

Next, we model the neural system as a recurrent neural network driven by our inputs

$$\frac{1}{\gamma}\dot{\mathbf{r}} = -\mathbf{r} + \mathbf{g}(A\mathbf{r} + B\mathbf{x} + \mathbf{d}), \quad (4.2)$$

where  $\mathbf{r}$  is a real-valued vector of  $N$  reservoir neuron states,  $A$  is an  $N \times N$  matrix of connections between neurons,  $B$  is an  $N \times M$  matrix of connections from the  $M$  inputs to the neurons,  $\mathbf{d}$  is an  $N \times 1$  bias vector,  $\mathbf{g}$  is a scalar activation function applied entry-wise to its input arguments (hence mapping vectors to vectors), and  $\gamma$  is a time constant. We use  $g = \tanh$  in the main text, and replicate many results using Wilson-Cowan oscillators in the Appendix in Chapter 5.

Several prior studies use echo state (Jaeger, 2010) and FORCE learning (Sussillo and Abbott, 2009) which allow reservoirs to predict a chaotic time series by modifying the inter-neuron connections. This modification can be accomplished by using the chaotic time series  $\mathbf{x}(t)$  to drive the reservoir, thereby generating the reservoir time series  $\mathbf{r}(t)$  (Fig. 24a,b). Here,  $\mathbf{x}(t)$  and  $\mathbf{r}(t)$  are  $M \times T$  and  $N \times T$  matrices, respectively, from numerically evolving the differential equations over  $T$  time steps. By solving for a simple  $M \times N$  readout matrix  $W$  that uses linear combinations of reservoir states to approximate the input by minimizing the matrix 2-norm (see the Appendix in Chapter 5)

$$W = \arg \min_W \|\mathbf{W}\mathbf{r}(t) - \mathbf{x}(t)\|_2, \quad (4.3)$$

the output  $\hat{\mathbf{x}}(t) = \mathbf{W}\mathbf{r}(t)$  mimics the input  $\mathbf{x}(t)$  (Fig. 24c). Finally, we close the feedback loop by substituting the output as the input to create the autonomous reservoir (Fig. 24d)

$$\frac{1}{\gamma}\dot{\mathbf{r}}' = -\mathbf{r}' + \mathbf{g}((A + BW)\mathbf{r}' + \mathbf{d}), \quad (4.4)$$

whose evolution projects to a Lorenz-shaped attractor as  $\mathbf{x}'(t) = \mathbf{W}\mathbf{r}'(t)$  (Fig. 24e). Hence, reservoirs sustain internal representations of complex temporal information by learning to autonomously evolve along a chaotic attractor from example inputs.

To study how reservoirs might perform computations by manipulating these representations, we use a modified framework (Sussillo and Abbott, 2009) to include a vector of control parameters  $\mathbf{c}$  that map to the reservoir neurons through matrix  $C$  to yield

$$\frac{1}{\gamma}\dot{\mathbf{r}} = -\mathbf{r} + \mathbf{g}(A\mathbf{r} + B\mathbf{x} + C\mathbf{c} + \mathbf{d}). \quad (4.5)$$

We drive the reservoir with the original  $\mathbf{x}_0(t)$  and modified  $\mathbf{x}_c(t)$  versions of the input at varying values of  $\mathbf{c}$  to generate the corresponding reservoir time series  $\mathbf{r}_0(t), \mathbf{r}_c(t)$ . By concatenating the time series along the time dimension for the inputs  $\mathbf{x}(t) = [\mathbf{x}_0(t), \mathbf{x}_1(t), \dots]$  and the reservoir states  $\mathbf{r}(t) = [\mathbf{r}_0(t), \mathbf{r}_1(t), \dots]$ , we train a readout matrix according to Eq. 4.3 (Fig. 24f–h). In what follows, we demonstrate that the feedback reservoir not only autonomously evolves about the input trajectory as a stable memory, but also learns to interpolate and extrapolate the modification of this memory far outside of the training range.

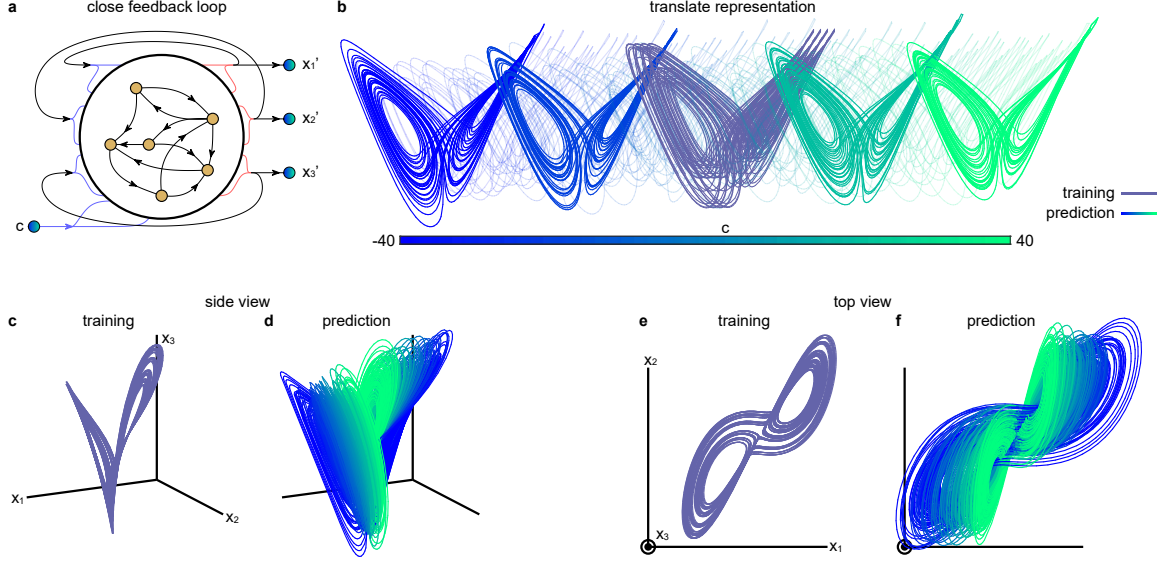
### 4.3. Learning a Translation Operation by Example

Reservoirs learn complex information through simple imitation: approximating the driving inputs using the reservoir states is enough to autonomously represent and evolve about a chaotic manifold. Here we show that this simple scheme is also enough to learn to translate and transform the representation.

We begin with a Lorenz time series  $\mathbf{x}_0(t)$ , and create shifted copies

$$\mathbf{x}_c(t) = \mathbf{x}_0(t) + P\mathbf{c}. \quad (4.6)$$

For the purposes of demonstration, we consider a translation in the  $x_1$  direction such that  $P = [1; 0; 0]$  is a column vector, and  $\mathbf{c} = 0, 1, 2, 3$  is a scalar. We use these four time series to drive our reservoir according to Eq. 4.5, thereby generating four reservoir time series  $\mathbf{r}_c(t)$ . We concatenate these series into  $\mathbf{r}(t)$  and  $\mathbf{x}(t)$ , and compute output weights according to



**Figure 25: Learning & Extrapolating Translations & Transformations by Example.** (a) Schematic of a feedback reservoir where the outputs  $W\mathbf{r}(t)$  replace the inputs  $\mathbf{x}(t)$  to create a closed feedback loop, where  $c$  remains a changeable parameter. (b) When the reservoir is trained on four Lorenz time series translated in the  $x_1$  dimension (purple), the feedback reservoir evolves autonomously about a Lorenz-shaped manifold, and translates this representation along  $x_1$  over the course of one simulation by smoothly and continuously changing  $c$  as a real number over a range much larger than the training range. (c,e) When the reservoir is trained on four linearly transformed (squeeze in the  $x_1$  direction, purple) time series, (d,f) the feedback reservoir similarly interpolates and extrapolates the transformation of its representation.

Eq. 4.3, such that our output  $\hat{\mathbf{x}} = W\mathbf{r}(t)$  approximates our input  $\mathbf{x}(t)$  (Fig. 24f–h). Finally, we substitute the output as the input to yield the feedback system (Fig. 25a)

$$\frac{1}{\gamma}\dot{\mathbf{r}}' = -\mathbf{r}' + \mathbf{g}(R\mathbf{r}' + C\mathbf{c} + \mathbf{d}), \quad (4.7)$$

where  $R = A + BW$ . As we evolve this autonomous reservoir while varying  $c$  to extreme values  $-40 \leq c \leq 40$  both inside and outside of the training values, it has learned to evolve about a Lorenz-shaped manifold that is translated based on the value of  $c$  (Fig. 25b, see the Appendix in Chapter 5 for more examples and Wilson-Cowan implementation).

We use the same scheme to teach reservoirs linear transformations. We begin with the

Lorenz time series  $\mathbf{x}_0(t)$  and create linearly transformed copies of the time series such that

$$\mathbf{x}_c(t) = (I + cP)\mathbf{x}_0(t), \quad (4.8)$$

for  $c = 0, 1, 2, 3$ , where  $P$  is a matrix encoding a transformation (Fig. 25c,e). Specifically, we perform a squeeze along  $x_1$  by setting  $[P]_{11} = -0.012$  and the remaining elements to 0. Exactly as before, we drive the reservoir according to Eq. 4.5, concatenate our input and reservoir time series into  $\mathbf{x}(t)$  and  $\mathbf{r}(t)$  to train the output weights  $W$  according to Eq. 4.3, and feed the outputs back as inputs to yield the feedback system Eq. 4.7. This reservoir autonomously evolves about a Lorenz-shaped manifold that stretches based on the parameter  $-40 \leq c \leq 40$  far outside of the parameters used in the training regime  $c = 0, 1, 2, 3$  (Fig. 25d,f: see the Appendix in Chapter 5 for more examples and Wilson-Cowan implementation). Hence, by training the network on translated and linearly transformed copies of the input, the reservoir has learned translation and linear transformation operations on the attractor.

#### 4.4. Learning to Infer Bifurcation Structure by Example

Next, we demonstrate that a reservoir can infer, without actually ever having observed, a much more dramatic change: a bifurcation. The first bifurcation occurs in the typical parameter regime (Eq. 4.1 for  $\rho \approx 28, \sigma = 10, \beta = 8/3$ ). Here, the Lorenz system contains two fixed points,  $\mathbf{x}_\rho^{*a}$  and  $\mathbf{x}_\rho^{*b}$ , that undergo a subcritical Hopf bifurcation when  $\rho = \rho^* \approx 24.7$  (Strogatz, 1994). When  $\rho < \rho^*$ , these two fixed points are stable. When  $\rho > \rho^*$ , the fixed points become unstable, yielding the characteristic wing-shaped flow.

To infer the bifurcation, we drive the reservoir with four training trajectories:  $\mathbf{x}_{23}^a(t)$  and  $\mathbf{x}_{23}^b(t)$  that evolve stably towards the fixed points for  $\rho = 23$  while  $c = 0$ , and  $\mathbf{x}_{24}^a(t)$  and  $\mathbf{x}_{24}^b(t)$  that evolve stably towards the fixed points for  $\rho = 24$  while  $c = 1$  (Fig. 26a). Importantly, we stress that the inputs include the transient spiral trajectory towards the fixed point. By training the output weights and evolving the feedback reservoir while

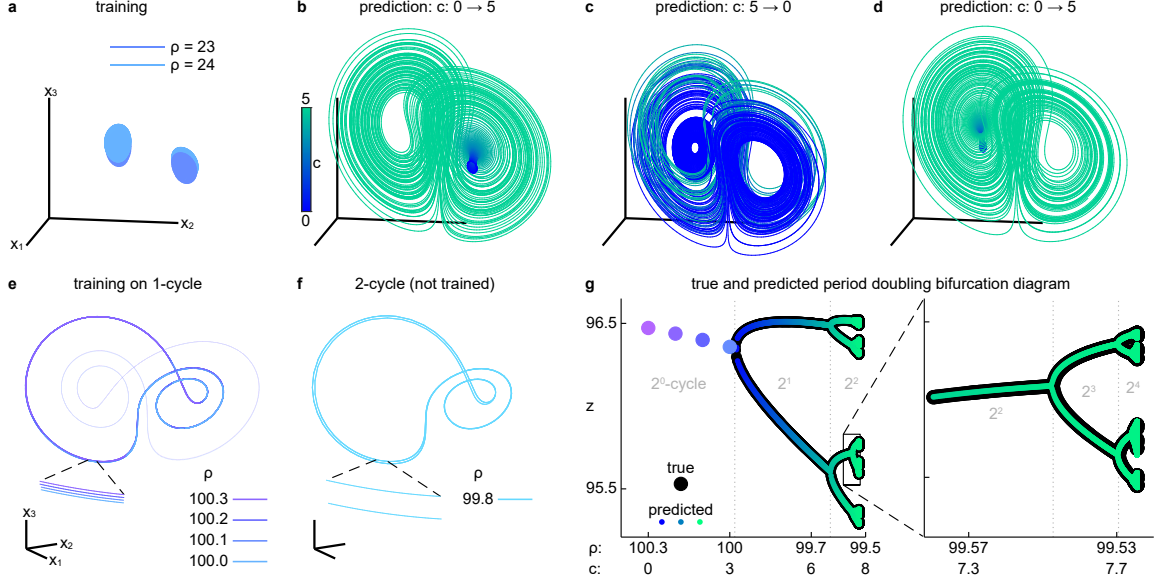


Figure 26: **Inferring & Extrapolating the Bifurcation of the Lorenz.** (a) Two training trajectories for each of the stable Lorenz fixed points at the wings, for  $\rho = 23$  with  $c = 0$  (blue) and for  $\rho = 24$  with  $c = 1$  (light blue). (b) The predicted trajectory of the feedback reservoir moves towards a stable fixed point for  $c = 0$ , and bifurcates into a Lorenz-shaped manifold as  $c$  is increased to 5. (c) As  $c$  is decreased back to 0, the predicted trajectory eventually falls into a stable fixed point. (d) As  $c$  is increased back to 5, the fixed point loses stability. (e) Four training 1-cycle trajectories for each wing (left wing: dark, right wing: light) of the Lorenz at  $\rho = 100.3, 100.2, 100.1$ , and  $100$ . (f) Example of a Lorenz bifurcation into a 2-cycle at  $\rho = 99.8$  that is not used in training. (g) True (black) and predicted (blue to green) Poincaré sections of the true Lorenz and predicted reservoir time series along the plane  $x = 0$  for  $\dot{x} > 0$ .

changing  $c$  from 0 to 5, the trajectory bifurcates into a Lorenz-shaped attractor (Fig. 26b). By subsequently changing  $c$  back to 0, the reservoir displays *transient chaos* whereby it temporarily evolves about a Lorenz-shaped attractor, but eventually falls into a stable fixed point (Fig. 26c). Finally, by changing  $c$  back to 5, the fixed points again become unstable (Fig. 26d).

In addition to inferring the Lorenz system's bifurcation and chaotic manifold geometry, reservoirs can accurately infer more detailed features of the Lorenz system's period doubling route to chaos in the parameter regime  $\rho \approx 100, \sigma = 10, \beta = 8/3$ . For  $\rho \gtrsim 99.98$ , the Lorenz evolves about a stable 1-cycle (Fig. 26e). Between  $99.95 \gtrsim \rho \gtrsim 99.63$ , the Lorenz bifurcates into a stable 2-cycle (Fig. 26f). As we continue to decrease  $\rho$ , the number of



cycles doubles until the flow becomes chaotic. We capture the period doubling phenomenon using a *Poincaré section* along the plane  $x = 0$  for  $\dot{x} > 0$  (see the Appendix in Chapter 5 for details, and Ref. (Langer and Parlitz, 2004) for the general modeling of parameter dependencies from time series.).

In this demonstration, we drive the reservoir with eight 1-cycle trajectories comprising four values of  $\rho$  at both wings (Fig. 26e), and perform feedback. As we increase  $c$ , the reservoir undergoes a series of period doubling bifurcations that almost exactly trace the true Lorenz system as measured by their Poincaré sections (Fig. 26g). Hence, we teach reservoirs to quantitatively infer highly nonlinear, latent, and unobserved bifurcations.

#### 4.5. Mechanism of how Operations are Learned

Now that we have taught reservoirs to manipulate chaotic manifolds, we seek to understand the mechanism by explicitly relating the change in the control input,  $d\mathbf{c}$ , to the change in reservoir trajectory,  $d\mathbf{r}'(t)$ . We consider the time series  $\mathbf{r}'(t) = \mathbf{r}'_{c=0}(t)$  generated by evolving the autonomous reservoir according to Eq. 4.7 at  $\mathbf{c} = \mathbf{0}$ . Next, we take the total differential of Eq. 4.7 evaluated at  $\mathbf{r}'(t)$  and  $\mathbf{c} = \mathbf{0}$  to yield

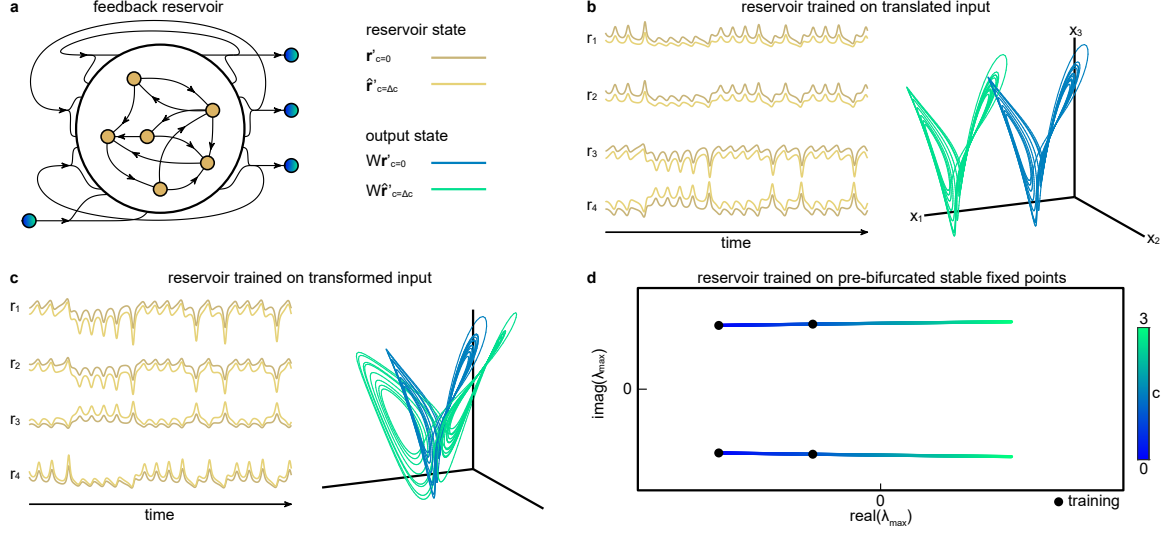
$$(I - KA)d\mathbf{r}' + \frac{1}{\gamma}d\dot{\mathbf{r}}' = K(BWd\mathbf{r}' + Cd\mathbf{c}), \quad (4.9)$$

where  $K = \text{diag}(d\mathbf{g}_{\mathbf{r}'=\mathbf{r}'(t), \mathbf{c}=\mathbf{0}})$ , and aim to write  $d\mathbf{r}'(t)$  as a function of  $d\mathbf{c}$ .

When learning translations, the output weights are trained such that  $W\mathbf{r}_c(t) \approx \mathbf{x}_c(t) = \mathbf{x}(t) + P\mathbf{c}$ . For sufficiently nearby training examples (small  $P, \mathbf{c}$ ), we also implicitly approximate the differential relation  $Wd\mathbf{r}(t) \approx Pd\mathbf{c}$ . Additionally, if the feedback reservoir stabilizes these examples, then  $Wd\mathbf{r}'(t) \approx Pd\mathbf{c}$ . Substituting this relation into Eq. 4.9 yields

$$(I - KA)d\mathbf{r}' + \frac{1}{\gamma}d\dot{\mathbf{r}}' \approx K(BP + C)d\mathbf{c}. \quad (4.10)$$

If we fix  $d\mathbf{c}$ , we have  $2N$  variables,  $d\mathbf{r}'$  and  $d\dot{\mathbf{r}}'$ , but only  $N$  equations. By taking the time



**Figure 27: Changing the Control Parameter Changes the Reservoir Dynamics to Manipulate Representations.** (a) Schematic of a reservoir with feedback connections after the output weights  $W$  have been trained. (b) Reservoir time series generated by evolving the autonomous reservoir with the original Lorenz input with  $c = 0$  (dark gold). We also show the predicted time series from solving Eq. 4.12 after training on translated examples and computing  $\Delta c \frac{dr'}{dc}$ , where  $\Delta c = 20$  (light gold). The output projections of the two time series are shown in blue and green, respectively. (c) The original and predicted reservoir states and their output projections for  $\Delta c = -40$  after training on transformed Lorenz inputs by solving Eq. 4.13. (d) Plot of the real and imaginary components of the two most unstable eigenvalues of the autonomous reservoir trained on two stable Lorenz trajectories (Fig. 26a). The reservoir is linearized about its equilibrium point  $\mathbf{r}_c^*$  as we change  $c^*$ .

derivative of the differential relation, we generate another  $N$  variables and  $N$  equations. Continuing to take time derivatives yields the following system of equations

$$\begin{bmatrix} H_0 & H_{-1} & 0 & \cdots \\ H_1 & H_0 & H_{-1} & \cdots \\ H_2 & 2H_1 & H_0 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} d\mathbf{r}' \\ d\dot{\mathbf{r}}' \\ d\ddot{\mathbf{r}}' \\ \vdots \end{bmatrix} \approx \begin{bmatrix} K \\ \dot{K} \\ \ddot{K} \\ \vdots \end{bmatrix} (BP + C)d\mathbf{c}, \quad (4.11)$$

where  $H_{-1} = \frac{1}{\gamma}I$ ,  $H_0 = I - KA$ , and  $H_i = -K^{(i)}A$  is the  $i$ -th time-derivative of  $KA$ . This matrix is a block-Hessenberg matrix, with an analytic solution (Słowiak, 2018) for the first term  $d\mathbf{r}'$ . We truncate this solution (see the Appendix in Chapter 5) to explicitly relate  $d\mathbf{r}'$

to  $d\mathbf{c}$  as follows:

$$d\mathbf{r}' \approx - \left[ \gamma H_0^2 - H_1 \right]^{-1} \begin{bmatrix} -\gamma H_0 & I \end{bmatrix} \begin{bmatrix} K \\ \dot{K} \end{bmatrix} (BP + C)d\mathbf{c}. \quad (4.12)$$

As a demonstration, we pick a finite  $\Delta\mathbf{c} = 20$ , and plot the original and predicted change in the reservoir states, and their outputs in spatial coordinates (Fig. 27b). Hence, using only the feedback dynamics Eq. 4.7 and sufficiently nearby training examples, changing  $\mathbf{c}$  causes changes in the reservoir states from Eq. 4.12 that map to a translation.

The same approach can be used for linear transformations, where the output weights are trained such that  $W\mathbf{r}_c(t) \approx \mathbf{x}_c(t) = (I + cP)\mathbf{x}(t)$ . For sufficiently nearby training examples, we implicitly approximate the differential relation  $Wd\mathbf{r}(t) \approx P\mathbf{x}(t)d\mathbf{c} \approx PW\mathbf{r}(t)d\mathbf{c}$ , which if properly stabilized, yields  $Wd\mathbf{r}'(t) \approx PW\mathbf{r}'(t)$ . Performing the same time derivatives and solution truncation as in the translation, we get the following relation between  $d\mathbf{c}$  and  $d\mathbf{r}'$ :

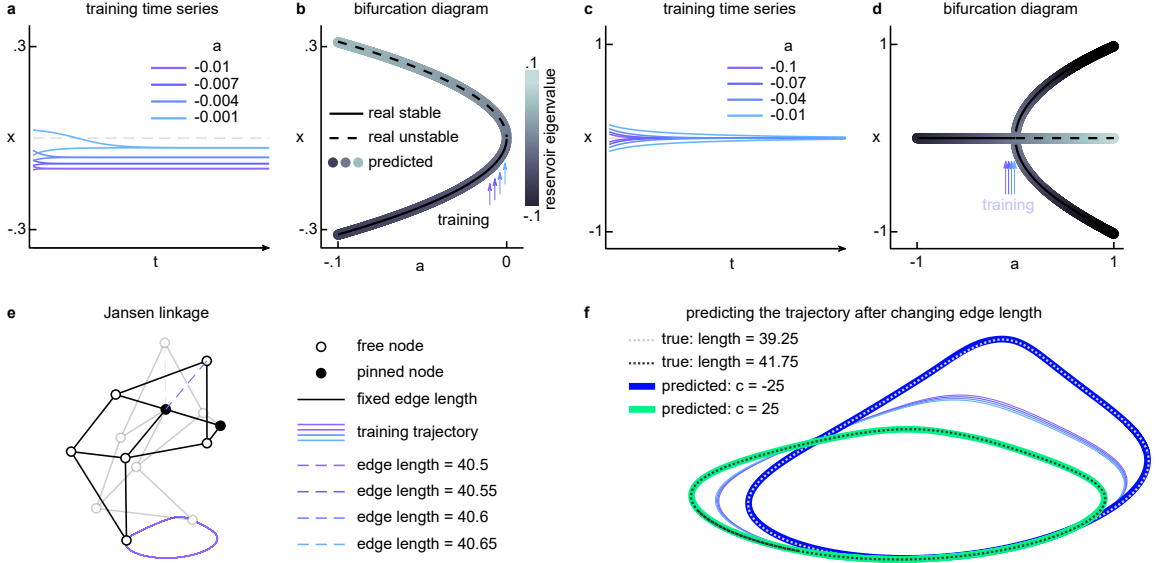
$$d\mathbf{r}' \approx - \left[ \gamma H_0^2 - H_1 \right]^{-1} \begin{bmatrix} -\gamma H_0 & I \end{bmatrix} \begin{bmatrix} K(BPW\mathbf{r}' + C) \\ \dot{K}(BPW\mathbf{r}' + C) + KBPW\dot{\mathbf{r}}' \end{bmatrix} d\mathbf{c}. \quad (4.13)$$

As another demonstration, we set  $\Delta\mathbf{c} = -40$ , and plot the original and predicted change in the reservoir states, and their outputs (Fig. 27c).

Finally, to understand how the reservoir is able to infer a bifurcation, we demonstrate that it learns a smooth translation of eigenvalues. Specifically, at  $\rho^*$ , the fixed points at the wings of the Lorenz system undergo a Hopf bifurcation, whereby the real component of complex conjugate eigenvalues goes from negative to positive. To track the eigenvalues of the autonomous reservoir, we linearize Eq. 4.7 about the equilibrium point  $\mathbf{r}_c^*, \mathbf{c}^*$  (see the Appendix in Chapter 5). Then, using the output weights trained only on stable Lorenz trajectories (at  $c = 0, \rho = 23$  and  $c = 1, \rho = 24$ ; Fig. 26a,b), we track the autonomous reservoir's two most unstable eigenvalues (largest real component) at the fixed point as we vary the control parameter from  $c = 0$  to  $c = 3$ . We find that these eigenvalues are

complex conjugates whose real components go from negative to positive (Fig. 27d). Hence, we demonstrate that not only can reservoirs learn smooth translations and transformations by mapping  $d\mathbf{c}$  to  $d\mathbf{r}'$ , but they can also perform bifurcations by learning smooth changes in their eigenvalues.

#### 4.6. Bifurcation Normal Forms & Non-Dynamical Time Series



**Figure 28: Inferring Bifurcation Normal Forms & Extrapolating Kinematic Trajectories.** (a) Training time series of the saddle-node normal form, where  $c = 0$  for  $a = -0.01$ ,  $c = 1$  for  $a = -0.007$ ,  $c = 2$  for  $a = -0.004$ , and  $c = 3$  for  $a = -0.001$ . (b) Bifurcation diagram of both the saddle-node normal form as we vary  $a$ , and the projection of reservoir fixed points  $W\mathbf{r}_c^*$  colored by the largest real eigenvalue of the reservoir Jacobian at  $\mathbf{r}_c^*$  as we vary  $c$ . (c) Training time series of the supercritical pitchfork normal form, where  $c = 0$  for  $a = -0.1$ ,  $c = 1$  for  $a = -0.07$ ,  $c = 2$  for  $a = -0.04$ , and  $c = 3$  for  $a = -0.01$ , along with the (d) corresponding real and predicted bifurcation diagram. (e) Jansen linkage that is pinned at the black nodes, leaving one degree of freedom whereby the bottom joint traces a cyclic trajectory. (f) Training (light blue to purple) and testing (dotted lines) trajectories of the true Jansen linkage by varying the dashed link, and the predicted reservoir trajectories (blue, green).

To demonstrate the generalizability of these principles, we teach reservoirs to infer the bifurcation diagrams of other dynamical normal forms, and to extrapolate non-dynamical

time series data. We begin with a saddle-node bifurcation with dynamical normal form

$$\dot{x} = a + x^2, \quad (4.14)$$

where  $a$  is the bifurcation parameter. When  $a < 0$ , the system has a stable fixed point at  $-\sqrt{-a}$  and an unstable fixed point at  $\sqrt{-a}$ . When  $a > 0$ , the system has no fixed points. We first generate two time series at each of four values of  $a < 0$ : one above the stable fixed point, and one below (Fig. 28a). We then drive the reservoir with these inputs while varying  $c$ , train the output weights, and track the feedback reservoir’s fixed points and largest real eigenvalue component as we change the control parameter  $c$  (see the Appendix in Chapter 5). We observe that the feedback reservoir’s fixed points almost exactly trace both the position and stability of the real saddle-node normal form as we vary  $a$  (Fig. 28b).

Next, we consider the supercritical pitchfork bifurcation with normal form

$$\dot{x} = ax - x^3. \quad (4.15)$$

When  $a < 0$ , Eq. 4.15 has one stable fixed point at  $x = 0$ . When  $a > 0$ , Eq. 4.15 has two stable fixed points at  $\pm\sqrt{a}$ , and one unstable fixed point at  $x = 0$ . We again generate two time series at each of four values of  $a < 0$ : one where  $x(t) > 0$ , and another where  $x(t) < 0$  (Fig. 28c). We then drive the reservoir while varying  $c$ , train the output weights, and again observe that the feedback reservoir’s fixed points almost exactly trace both the position and stability of the pitchfork normal form’s fixed points (Eq. 4.15, Fig. 28d). We emphasize that in both normal forms, the reservoir was able to infer the location and stability of all fixed points after only observing a few trajectories to one of the stable fixed points.

Finally, we test whether the reservoir’s ability to extrapolate generalizes to non-dynamical time series. Specifically, we consider the kinematic trajectory of a modified version of the Jansen linkage (Nansai et al., 2013), whose bottom joint traces out a cyclical walking path with 1 degree of freedom (Fig. 28e). As the length of the links change, the geometry

of the traced trajectory also changes in a complex manner. Additionally, the evolution of the original and changed trajectories cannot be written as a dynamical system of the form  $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ . We first generate four training trajectories by changing the length of the dashed link by  $\Delta l = 0.05$  while varying  $c$  by  $\Delta c = 1$  (Fig. 28e,f), and perform training and feedback. Then, we use the feedback reservoir to generate predicted trajectories at  $c = -25$  and  $c = 25$ , and observe that they very closely follow the true trajectories when the link length changes by  $-25\Delta l$  and  $25\Delta l$  (Fig. 28f).

#### 4.7. Simultaneous Learning of Multiple Operations

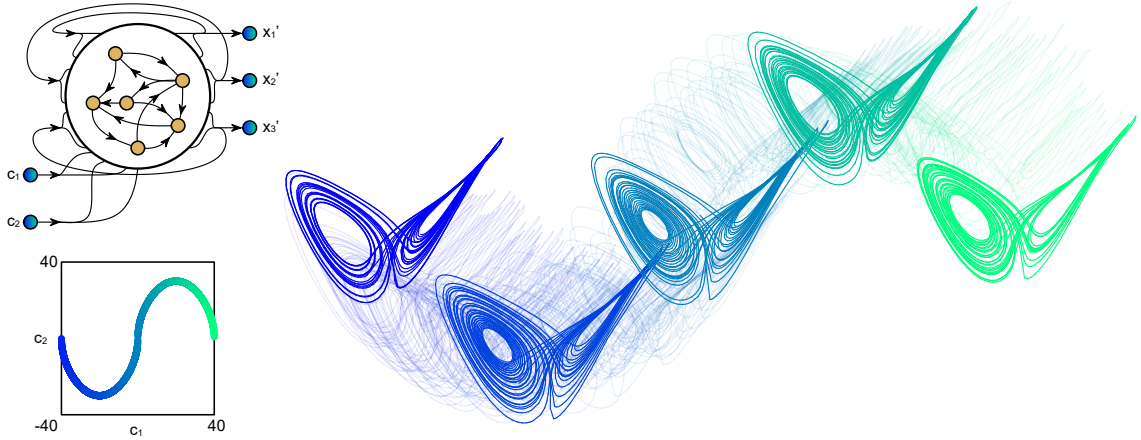


Figure 29: **Flight of the Lorenz.** A reservoir trained on translated inputs along the  $x_1$  and  $x_3$  directions evolves autonomously along a Lorenz-shaped chaotic manifold. We can change the  $x_1$  and  $x_3$  position of its representation by changing control parameters  $c_1$  and  $c_2$ , respectively.

To close, here we demonstrate that reservoirs can easily learn multiple computations by changing multiple control inputs. We train a translation in the  $x_1$  direction with control parameter  $c_1$ , and a translation in the  $x_3$  direction with control parameter  $c_2$ . As before, we begin with a Lorenz time series  $\mathbf{x}_{0,0}(t)$  generated from Eq. 4.1, and created shifted copies

$$\mathbf{x}_{c_1, c_2}(t) = \mathbf{x}_{0,0}(t) + c_1 \mathbf{a}_1 + c_2 \mathbf{a}_2, \quad (4.16)$$

where  $\mathbf{a}_1 = [1; 0; 0]$  corresponds to an  $x_1$  shift, and  $\mathbf{a}_2 = [0; 0; 1]$  corresponds to an  $x_3$  shift.

We generate 10 shifted inputs, with one unshifted attractor ( $c_1 = 0, c_2 = 0$ ), three shifts in the  $x_1$  direction ( $c_1 = 1, 2, 3, c_2 = 0$ ), three shifts in the  $x_3$  direction ( $c_1 = 0, c_2 = 1, 2, 3$ ), and three shifts in both directions ( $c_1 = 1, 2, 3, c_2 = 1, 2, 3$ ). We use these shifted copies along with their corresponding control inputs to drive our reservoir and produce 10 reservoir time series  $\mathbf{r}_{c_1, c_2}(t)$ . Then, we concatenate these 10 time series into  $\mathbf{x}(t)$  and  $\mathbf{r}(t)$  to train output weights  $W$  according to Eq. 4.3, and perform the feedback according to Eq. 4.7 where  $\mathbf{c} = [c_1; c_2]$  is a vector. By changing parameters  $c_1$  and  $c_2$ , the reservoir evolves about a Lorenz-shaped manifold that is shifted in the  $x_1$  and  $x_3$  directions (Fig. 29).

#### 4.8. Discussion

In this paper, we teach an RNN to interpolate, extrapolate, and infer global bifurcation structures and manipulations after observing only a few local exemplars. Our approach contributes to prior work on artificial neural networks in three significant ways (Seung, 1998; Wu et al., 2016; Jaeger, 2010; Sussillo and Abbott, 2009; Klos et al., 2020). First, we provide a means by which a neural system performs accurate and extreme interpolations and extrapolations of modifications to its own representation far outside of its training regime. Second, we provide an analytic mechanism by which this meta-learning occurs, thereby providing design principles for effective teaching (e.g. small  $c$ , closely spaced exemplars). Finally, we demonstrate that neural systems can infer global and highly nonlinear bifurcation structure using only local, pre-bifurcated example trajectories in many systems. Importantly, we use a randomly generated network that does not need to be artificially engineered to preserve invariance or manipulate information (Wu et al., 2016).

One of the main limitations of this work is the lack of a clear mechanism of how the network connectivity ultimately stabilizes the chaotic manifold. Much progress has been made in tackling this limitation, both by exercising theoretical concepts of generalized synchronization (Rulkov et al., 1995b), and by developing tools for controlling chaos (Ott et al., 1990). However, there is insufficient knowledge to guarantee that a set of training

and reservoir parameters will always successfully teach the desired computation. Similarly, we are unable to specify exactly how far to space the training examples for the feedback reservoir to successfully learn the linear relationships between the differential of the reservoir states and the control parameter.

A particularly promising area for future work is related to the generalizability of the learning mechanism across activation functions, as demonstrated by the replication of our results in the more nonlinear Wilson-Cowan network model (Wilson and Cowan, 1972). Understanding the role of various types and degrees of nonlinearity in learning computations may provide insights for the intervention and design of biological and artificial neural systems, respectively. Additionally, these results provide a basis for exploring more complex computations, such as inferring bifurcations in experimental data, and testing the reservoir’s “imagination” in reconstructing more complex chaotic manifolds using incomplete data. Finally, and perhaps most astonishing is the reservoir’s ability to accurately reconstruct the global nonlinear geometry of both the bifurcated Lorenz manifold and various bifurcation diagrams, as well as the complex changes in kinematic trajectories after only observing a narrow range of pre-bifurcated or kinematic examples. The accurate reconstruction implies that the reservoir is actually inferring higher-order nonlinear structure. This work therefore provides a starting point for exploring exactly how higher-order structure is learned by neural systems.



## CHAPTER 5 : Appendix to Teaching Recurrent Neural Networks to Infer Global Structure

In this appendix, we describe additional details about the methods and simulations used in the main text. We begin with a more thorough overview of reservoir dynamics and their derivation, followed by specific details of the numerical simulations.

### 5.1. Reservoir Dynamics

The reservoir computing framework is a general scheme by which a nonlinear dynamical system (the reservoir) is driven by some input, and a simple linear readout of the reservoir states is trained. The reservoir consists of  $N$  neural units, where each unit  $i$  has a real-valued level of activity over time,  $r_i(t)$ . We collect this activity into an  $N$ -dimensional column vector

$$\mathbf{r}(t) = \begin{bmatrix} r_1(t) \\ r_2(t) \\ \vdots \\ r_N(t) \end{bmatrix}, \quad (5.1)$$

that we refer to as the *reservoir state*. These reservoir states are driven by some input time series of  $M$  inputs  $x_1(t), x_2(t), \dots, x_M(t)$ , that we collect into the input vector

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_M(t) \end{bmatrix}. \quad (5.2)$$

In our framework, we add a set of  $K$  control inputs  $c_1, \dots, c_K$  that we collect into the control vector

$$\mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_K \end{bmatrix}. \quad (5.3)$$

For continuous time systems ( $t \in \mathbb{R}_{\geq 0}$ ), a typical equation for the time-evolution of a reservoir consists of a nonlinear (usually sigmoidal) transformation  $\mathbf{g}$  on a linear sum of all inputs and states written as

$$\frac{1}{\gamma} \dot{\mathbf{r}}(t) = -\mathbf{r}(t) + \mathbf{g}(A\mathbf{r}(t) + B\mathbf{x}(t) + C\mathbf{c} + \mathbf{d}), \quad (5.4)$$

where  $\dot{\mathbf{r}}(t)$  represents the time derivative,  $A$  is a real-valued matrix of dimension  $N \times N$ ,  $B$  is a real-valued matrix of dimension  $N \times M$ ,  $C$  is a real-valued matrix of dimension  $N \times K$ , and  $\mathbf{d}$  is a constant bias vector of dimension  $N \times 1$ . We can write the dynamics for each reservoir state,  $r_i(t)$ , as

$$\frac{1}{\gamma} \dot{r}_i(t) = -r_i(t) + g_i \left( \sum_{n=1}^N A_{in} r_n(t) + \sum_{m=1}^M B_{im} x_m(t) + \sum_{k=1}^K C_{ik} c_k + d_i \right). \quad (5.5)$$

If we write  $A_{i*}$ ,  $B_{i*}$ , and  $C_{i*}$  as the  $i$ -th row of matrices  $A$ ,  $B$ , and  $C$ , respectively, we can write this equation more concisely as

$$\frac{1}{\gamma} \dot{r}_i(t) = -r_i(t) + g_i (A_{i*} \mathbf{r}(t) + B_{i*} \mathbf{x}(t) + C_{i*} \mathbf{c} + d_i). \quad (5.6)$$

We begin by observing that the reservoir states are evolved according to some predetermined input  $\mathbf{x}(t)$  and control input  $\mathbf{c}$  to generate the reservoir state time series  $\mathbf{r}(t)$ . Next, linear combinations of the reservoir state are taken to approximate the input  $\mathbf{x}(t)$  by minimizing

the matrix 2-norm of the difference in the numerical time series (see Sec. 5.9)

$$\|W\mathbf{r}(t) - \mathbf{x}(t)\|_2, \quad (5.7)$$

where  $W$  is the real valued matrix of dimension  $M \times N$  that is trained. After training, we perform feedback by replacing the inputs  $\mathbf{x}(t)$  with the trained outputs  $W\mathbf{r}(t)$  to yield the feedback dynamics

$$\frac{1}{\gamma}\dot{\mathbf{r}}'(t) = -\mathbf{r}'(t) + \mathbf{g}(A\mathbf{r}'(t) + BW\mathbf{r}'(t) + C\mathbf{c} + \mathbf{d}), \quad (5.8)$$

and by factoring the term  $R = A + BW$ , we obtain

$$\frac{1}{\gamma}\dot{\mathbf{r}}'(t) = -\mathbf{r}'(t) + \mathbf{g}(R\mathbf{r}'(t) + C\mathbf{c} + \mathbf{d}). \quad (5.9)$$

This feedback equation is written element-wise as

$$\frac{1}{\gamma}\dot{r}'_i(t) = -r'_i(t) + g_i \left( \sum_{n=1}^N R_{in}r'_n(t) + \sum_{k=1}^K C_{ik}c_k + d_i \right). \quad (5.10)$$

We note that  $R$  is an  $N \times N$  matrix.

## 5.2. Form of Dynamical Equations

In this work, we implement two different neural models: the hyperbolic tangent and the Wilson-Cowan. For convenience, we write all inputs into neuron  $i$  as

$$z_i(\mathbf{r}, \mathbf{x}, \mathbf{c}) = \sum_{n=1}^N A_{in}r_n(t) + \sum_{m=1}^M B_{im}x_m(t) + \sum_{k=1}^K C_{ik}c_k + d_i. \quad (5.11)$$

Then, the dynamical equations of each hyperbolic tangent unit can be written as

$$\frac{1}{\gamma}\dot{r}_i(t) = -r_i(t) + \tanh(z_i(\mathbf{r}, \mathbf{x}, \mathbf{c})), \quad (5.12)$$

and the equations of all units can be concisely written as

$$\frac{1}{\gamma}\dot{\mathbf{r}} = -\mathbf{r} + \tanh(\mathbf{z}), \quad (5.13)$$

where  $\mathbf{z}$  is the vector of collected inputs.

Alternatively, the Wilson-Cowan (WC) oscillator dynamics are slightly more complex, consisting of an  $N/2$ -dimensional vector of activity for the excitatory populations,  $\mathbf{E}$ , and an equivalently sized  $N/2$ -dimensional vector of activity for the inhibitory populations,  $\mathbf{I}$ . Let  $A_{EE}$ ,  $A_{EI}$ ,  $A_{IE}$ , and  $A_{II}$  be non-negative  $N/2 \times N/2$  matrices defining weights from  $\mathbf{E} \rightarrow \mathbf{E}$ ,  $\mathbf{E} \rightarrow \mathbf{I}$ ,  $\mathbf{I} \rightarrow \mathbf{E}$ , and  $\mathbf{I} \rightarrow \mathbf{I}$ , respectively. Further, let the input matrices  $B_E$  and  $B_I$  project from the inputs  $\mathbf{x}(t)$  into the excitatory and inhibitory populations, respectively, and  $C_E$  and  $C_I$  project from the control inputs  $\mathbf{c}$  to the excitatory and inhibitory populations, respectively. For convenience we collect all inputs into the excitatory population as

$$\mathbf{z}_E = A_{EE}\mathbf{E} - A_{IE}\mathbf{I} + B_E\mathbf{x} + C_E\mathbf{c} + \mathbf{d}_E, \quad (5.14)$$

and all inputs into the inhibitory population as

$$\mathbf{z}_I = A_{EI}\mathbf{E} - A_{II}\mathbf{I} + B_I\mathbf{x} + C_I\mathbf{c} + \mathbf{d}_I, \quad (5.15)$$

where  $\mathbf{d}_E$  and  $\mathbf{d}_I$  are the bias terms for the excitatory and inhibitory populations, respectively. Then, the WC dynamics become

$$\frac{1}{\gamma}\dot{\mathbf{E}} = -\mathbf{E} + (1 - r_r\mathbf{E})\phi(\mathbf{z}_E), \quad (5.16)$$

and

$$\frac{1}{\gamma}\dot{\mathbf{I}} = -\mathbf{I} + (1 - r_r\mathbf{I})\phi(\mathbf{z}_I), \quad (5.17)$$

where  $r_r$  are derived from the time coarse-grained refractory periods, and  $\phi$  is a sigmoidal

activation function of the form

$$\phi(\mathbf{z}) = \frac{1}{1 + e^{-\mathbf{z}}}, \quad (5.18)$$

where the exponentiation and division is performed element-wise.

In these equations, the diagonal entries of  $A_{EE}$ ,  $A_{IE}$ ,  $A_{EI}$ , and  $A_{II}$  represent the connections between the excitatory and inhibitory populations of one canonical WC oscillator, while the non-diagonal entries represent the connections between oscillators. In this work, we assume that inputs reach both excitatory and inhibitory populations (i.e.  $B_E, B_I, C_E, C_I$  can be dense), that excitatory populations can project to both the excitatory and inhibitor populations of all oscillators (i.e.  $A_{EE}$  and  $A_{EI}$  can be dense), and that inhibitory population of any oscillator can only project to itself and the excitatory population of its own oscillator (i.e.  $A_{IE}$  and  $A_{II}$  are diagonal). If we compile the four connectivity and input matrices into one matrix and adjust the signs as

$$A = \begin{bmatrix} A_{EE} & -A_{IE} \\ A_{EI} & -A_{II} \end{bmatrix}, \quad B = \begin{bmatrix} B_E \\ B_I \end{bmatrix}, \quad C = \begin{bmatrix} C_E \\ C_I \end{bmatrix}, \quad (5.19)$$

and if we additionally order the excitatory and inhibitory populations as

$$\mathbf{r} = \begin{bmatrix} \mathbf{E} \\ \mathbf{I} \end{bmatrix}, \quad (5.20)$$

where the  $i$ -th entry of  $\mathbf{E}$  and  $\mathbf{I}$  corresponds to excitatory and inhibitory populations of oscillator  $i$ , then we can write the WC dynamics as

$$\frac{1}{\gamma} \dot{\mathbf{r}} = -\mathbf{r} + (1 - r_e \mathbf{r}) \phi(A\mathbf{r} + B\mathbf{x} + C\mathbf{c} + \mathbf{d}), \quad (5.21)$$

to yield a compact equational form.

### 5.3. Evaluation of the Jacobian

Whether to track fixed points or to evaluate linearized dynamics, we frequently evaluate the Jacobian of our reservoir system about some equilibrium point. Here we explain this process in more detail. We begin with the general reservoir dynamics

$$\frac{1}{\gamma}\dot{\mathbf{r}} = -\mathbf{r} + \mathbf{g}(A\mathbf{r} + B\mathbf{x} + C\mathbf{c} + \mathbf{d}), \quad (5.22)$$

and consider some equilibrium point  $\mathbf{r}^*, \mathbf{x}^*, \mathbf{c}^*$ , where

$$\mathbf{0} = -\mathbf{r}^* + \mathbf{g}(A\mathbf{r}^* + B\mathbf{x}^* + C\mathbf{c}^* + \mathbf{d}). \quad (5.23)$$

Next, we take the gradient of our general reservoir dynamics, and evaluate the Taylor series expansion to first order about the equilibrium point, such that

$$\frac{1}{\gamma}\dot{\mathbf{r}} \approx -(\mathbf{r} - \mathbf{r}^*) + \text{diag}(d\mathbf{g}_{\mathbf{r}=\mathbf{r}^*, \mathbf{x}=\mathbf{x}^*, \mathbf{c}=\mathbf{c}^*})[A(\mathbf{r} - \mathbf{r}^*) + B(\mathbf{x} - \mathbf{x}^*) + C(\mathbf{c} - \mathbf{c}^*)], \quad (5.24)$$

where  $d\mathbf{g}_{\mathbf{r}=\mathbf{r}^*, \mathbf{x}=\mathbf{x}^*, \mathbf{c}=\mathbf{c}^*}$  is the differential of the activation function  $\mathbf{g}$ . For example, if the activation function is

$$\mathbf{g}(\mathbf{r}^*, \mathbf{x}^*, \mathbf{c}^*) = \tanh(A\mathbf{r}^* + B\mathbf{x}^* + C\mathbf{c}^* + \mathbf{d}), \quad (5.25)$$

then the differential is

$$d\mathbf{g}_{\mathbf{r}=\mathbf{r}^*, \mathbf{x}=\mathbf{x}^*, \mathbf{c}=\mathbf{c}^*} = (\mathbf{1} - \tanh^2(A\mathbf{r}^* + B\mathbf{x}^* + C\mathbf{c}^* + \mathbf{d})), \quad (5.26)$$

where  $\mathbf{1}$  is a vector of 1s.

The process remains the same for the feedback dynamics

$$\frac{1}{\gamma}\dot{\mathbf{r}} = -\mathbf{r} + \mathbf{g}(R\mathbf{r} + C\mathbf{c} + \mathbf{d}), \quad (5.27)$$

where  $R = A + BW$  after feedback. We consider some equilibrium point  $\mathbf{r}^*, \mathbf{c}^*$  where

$$\mathbf{0} = -\mathbf{r}^* + \mathbf{g}(R\mathbf{r}^* + C\mathbf{c}^* + \mathbf{d}), \quad (5.28)$$

and evaluate the Taylor series expansion to first order

$$\frac{1}{\gamma} \dot{\mathbf{r}} = -(\mathbf{r} - \mathbf{r}^*) + \text{diag}(d\mathbf{g}_{\mathbf{r}=\mathbf{r}^*, \mathbf{c}=\mathbf{c}^*})[R(\mathbf{r} - \mathbf{r}^* + C(\mathbf{c} - \mathbf{c}^*))]. \quad (5.29)$$

#### 5.4. Bias Term & Equilibrium Point Selection

To ensure that our non-driven reservoir begins with stable dynamics, we initialize our reservoir parameters by first selecting a distribution of reservoir equilibrium points, and then selecting the bias term to achieve that equilibrium point. Specifically, we wish to ensure that at zero input (i.e.  $\mathbf{x}^* = \mathbf{0}$  and  $\mathbf{c}^* = \mathbf{0}$ ), the non-driven reservoir dynamics exist at some stable fixed point  $\mathbf{r}^*$ . To achieve this goal, we consider the reservoir equations at the desired fixed point,

$$\frac{1}{\gamma} \dot{\mathbf{r}} = \mathbf{0} = -\mathbf{r}^* + \mathbf{g}(A\mathbf{r}^* + \mathbf{d}), \quad (5.30)$$

and simply solve for  $\mathbf{d}$  as

$$\mathbf{d} = \mathbf{g}^{-1}(\mathbf{r}^*) - A\mathbf{r}^*. \quad (5.31)$$

For example, if  $\mathbf{g}$  is  $\tanh$ , then, the bias term becomes

$$\mathbf{d} = \tanh^{-1}(\mathbf{r}^*) - A\mathbf{r}^*. \quad (5.32)$$

#### 5.5. Tracking the Reservoir Fixed Points

Throughout the text, we track the fixed points of the feedback reservoir as we change the control parameter  $\mathbf{c}$ , such as when we generate the bifurcation diagrams. This process consists of two steps. First, we have to identify the feedback reservoir's fixed points at some value of  $\mathbf{c}$ . Second, we have to track that fixed point as we change  $\mathbf{c}$ .

For the first part of identifying the feedback reservoir's fixed points, consider the simple case where we drive the reservoir with a constant input  $\mathbf{x}^*$  at  $\mathbf{c} = \mathbf{c}^*$ , thereby generating a constant reservoir output  $\mathbf{r}^*$ . By virtue of the reservoir output not changing, we know that the values  $\mathbf{x} = \mathbf{x}^*$ ,  $\mathbf{c} = \mathbf{c}^*$ , and  $\mathbf{r} = \mathbf{r}^*$  serve as an equilibrium solution to the reservoir dynamics such that

$$\mathbf{0} = -\mathbf{r}^* + \mathbf{g}(A\mathbf{r}^* + B\mathbf{x}^* + C\mathbf{c}^* + \mathbf{d}). \quad (5.33)$$

If training is successful such that  $W\mathbf{r}^* \approx \mathbf{x}^*$ , then  $\mathbf{r}^*$  becomes a fixed point of the feedback dynamics

$$\mathbf{0} \approx -\mathbf{r}^* + \mathbf{g}((A + BW)\mathbf{r}^* + C\mathbf{c}^* + \mathbf{d}). \quad (5.34)$$

However, in all of the bifurcation diagram inference examples, we only trained on trajectories that evolved stably towards one of fixed points (e.g. stable branch  $a$  of the saddle-node normal form, thereby generating reservoir output  $\mathbf{r}_a^*$ ) to generate  $W$ . Hence, we have no simulation data about where the other reservoir state  $\mathbf{r}_b^*$  might be when driven by the unstable branch  $b$ . The solution to this problem is simple: after we perform training and generate  $W$ , we open the feedback loop and drive the reservoir with the unstable fixed point (e.g. unstable branch  $b$  of the saddle-node normal form). Crucially, we note that we do **not** perform any training on these unstable trajectories. We **only** use them to generate a guess,  $\hat{\mathbf{r}}_b^*$ , of what the unobserved reservoir fixed point might be. If the feedback reservoir has no fixed point near  $\hat{\mathbf{r}}_b^*$ , then the next step may not converge and yield a large error (see below), or may converge to a distant fixed point.

Now that we can generate guesses for where the reservoir fixed points are, we use a root-finding method (specifically, the Newton-Raphson method) to converge to the true feedback reservoir fixed point. Essentially, the fixed-point equation of the feedback reservoir,

$$\mathbf{h}(\mathbf{r}) = -\mathbf{r} + \mathbf{g}((A + BW)\mathbf{r} + C\mathbf{c}^* + \mathbf{d}), \quad (5.35)$$

is a set of nonlinear equations whose roots are the fixed points. Hence, we use our fixed point



guesses,  $\hat{\mathbf{r}}_i^*$ , as the initial condition of the optimization, and iterate the Newton-Raphson method to converge to the true fixed point  $\mathbf{r}_i^*$ . Given a point  $\hat{\mathbf{r}}_i^*$  that is close to the true fixed point, the update rule is

$$\hat{\mathbf{r}}_i^*(n+1) = \hat{\mathbf{r}}_i^*(n) - J_{\mathbf{h}}(\hat{\mathbf{r}}_i^*(n))^{-1} \mathbf{h}(\hat{\mathbf{r}}_i^*(n)), \quad (5.36)$$

where  $J_{\mathbf{h}}(\hat{\mathbf{r}}_i^*(n))$  is the Jacobian of  $\mathbf{h}$  evaluated at  $\hat{\mathbf{r}}_i^*(n)$ . To evaluate the goodness of the optimization, we use the quadratic product of the fixed point equation as a measure of error

$$\text{err}(\mathbf{r}^*) = \mathbf{h}(\hat{\mathbf{r}}_i^*(n))^{\top} \mathbf{h}(\hat{\mathbf{r}}_i^*(n)). \quad (5.37)$$

We note that in all of our examples, the error was less than  $10^{-20}$ .

Finally, to achieve the second part of tracking the fixed points as we change the control parameter  $\mathbf{c}$ , we note that the estimated guess of the feedback reservoir fixed points,  $\hat{\mathbf{r}}_i^*$ , as well as the true post-optimization fixed points,  $\mathbf{r}_i^*$ , are generated at a specific value of  $\mathbf{c}^*$ . Once the optimization is complete, we incrementally change  $\mathbf{c}^*$ , use the previous feedback reservoir fixed point  $\mathbf{r}_i^*$  as the initial condition for the new optimization, and rerun the Newton-Raphson method at the new value of  $\mathbf{c}^*$ .

## 5.6. Poincaré Sections & the Period Doubling Bifurcation Diagram

In the main text, we construct the period doubling bifurcation diagram of the Lorenz system at  $\rho \approx 100$  using a Poincaré section, which keeps track of the Lorenz trajectory as it passes through the section. The Lorenz system evolves in Euclidean space  $\mathbb{R}^3$ . For the purposes of this manuscript, the Poincaré section is a 2-dimensional plane that transversely intersects the trajectory of the Lorenz system. Specifically, we choose the plane  $x = 0$ . By keeping track of when the trajectory passes through  $x = 0$ , we obtain a simple and quantitative summary of the complex flow.

When  $\rho = 100$ , the Lorenz system evolves about a 1-cycle, such that for each period, the

trajectory passes from  $x > 0$  to  $x < 0$  once, and from  $x < 0$  to  $x > 0$  once. To distinguish between the two, we focus on the latter intersection where the trajectory passes from  $x < 0$  to  $x > 0$ . Another way to state these conditions is to say that we track the point  $\mathbf{x}_i$  where the Lorenz trajectory passes through  $x = 0$  when  $\dot{x} > 0$ . For this one cycle, every intersection is at the same point. However, when  $\rho = 99.8$ , the trajectory becomes a 2-cycle, and every other intersection is at the same point. As the bifurcation continues to generate  $2^k$ -cycles, every  $2^k$  intersection is at the same point.

To generate the period doubling bifurcation diagram, we run the true Lorenz system or reservoir prediction at some value of  $\rho$  or  $c$ , respectively, until the transient period is over and the trajectories have fallen onto the appropriate  $2^k$  cycle. Then, we track the points of intersection of the trajectory with  $x = 0, \dot{x} > 0$ , and plot the  $z$ -coordinate of these points of intersection as a function of either  $\rho$  or  $c$ .

## 5.7. Simulation Parameters

All simulations were performed using either the tanh equations

$$\frac{1}{\gamma}\dot{\mathbf{r}} = -\mathbf{r} + \tanh(A\mathbf{r} + \alpha B\mathbf{x} + \beta C\mathbf{c} + \mathbf{d}), \quad (5.38)$$

or the Wilson-Cowan (WC) model

$$\frac{1}{\gamma}\dot{\mathbf{r}} = -\mathbf{r} + \frac{1 - r_r\mathbf{r}}{1 + e^{-(A\mathbf{r} + \alpha B\mathbf{x} + \beta C\mathbf{c} + \mathbf{d})}}. \quad (5.39)$$

- The elements of  $B$  and  $C$  were chosen densely, uniformly, and randomly between  $-1$  and  $1$ .
- For the tanh equations, the adjacency matrix  $A$  was chosen to be 10% dense, with elements chosen randomly and uniformly between  $-1$  and  $1$ . Then,  $A$  was divided by its largest real-component eigenvalue, and multiplied by  $0.95$ .

- For the WC equations, the diagonals of  $A_{IE}$  and  $A_{II}$  were chosen randomly and uniformly between 0 and 1, and off-diagonal entries set to 0. The matrices  $A_{EE}$  and  $A_{EI}$  were chosen to be 10% dense with elements chosen randomly and uniformly between 0 and 1, followed by a replacement of all diagonal entries (both zero and non-zero) with random uniform elements between 0 and 1. Once these four matrices were assembled into  $A$ , it was divided by its largest real-component eigenvalue, and multiplied by 0.95.
- $r_r$  was chosen to be 0.2.
- The fixed points of the tanh reservoir were initialized as randomly and uniformly distributed between  $[-1, -0.8] \cup [0.8, 1]$ , and the appropriate bias  $\mathbf{d}$  was chosen. This selection ensures the fixed points are distributed 10% of the whole range above the minimum ( $-1$ ) and maximum ( $1$ ) output of tanh.
- The fixed points of the WC reservoir were initialized as randomly and uniformly distributed between  $[0, 0.1] \cup [\frac{11}{15}, \frac{12.5}{15}]$ , and the appropriate bias was chosen. This selection ensures the fixed points are distributed 10% of the whole range above the minimum ( $0$ ) and maximum ( $12.5/15$ ) output of the WC with  $r_r = 0.2$ .
- Each simulation has an associated time step,  $\Delta t$ , throwaway simulation time per example to forget the transient period,  $T_{\text{waste}}$ , and the training simulation time per example,  $T_{\text{train}}$ .
- Each simulation also has an associated time constant,  $\gamma$ , and number of reservoir neurons,  $N$ .

Simulation	type	$\Delta t$	$T_{\text{waste}}$	$T_{\text{train}}$	$N$	$\gamma$	$\alpha$	$\beta$
Introduction	tanh	0.001	20	200	450	100	0.008	0.004
Translate: Lorenz	tanh	0.001	20	200	450	100	0.008	0.004
Transform: Lorenz	tanh	0.001	20	200	450	100	0.008	0.004
Bifurcate: Lorenz: $\rho \approx 28$	tanh	0.001	20	200	450	100	0.008	0.004
Bifurcate: Lorenz: $\rho \approx 100$	tanh	0.0002	10	50	300	100	0.002	0.002
Bifurcate: saddle-node	tanh	0.001	20	200	50	10	0.5	0.001
Bifurcate: pitchfork	tanh	0.001	20	200	50	10	0.5	0.001
Jansen linkage	tanh	0.001	50	50	900	1	0.7	0.013
Flight of the Lorenz	tanh	0.001	20	200	450	100	0.004	0.002
Appendix: translate Lorenz	WC	0.001	20	200	600	40	0.005	0.001
Appendix: transform Lorenz	WC	0.001	20	200	600	40	0.005	0.001
Appendix: bifurcate Lorenz	WC	0.001	20	200	600	40	0.005	0.001
Appendix: translate multiple	tanh	0.001	20	200	450	100	0.008	0.004
Appendix: transform multiple	tanh	0.001	20	200	450	100	0.008	0.004

Table 1: Simulation parameters

## 5.8. Simulation Method

To simulate both the input and reservoir dynamics, we used a 4-th order Runge-Kutta numerical integration. For the dynamics of the Lorenz attractor,

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}),$$

the Runge-Kutta computes the following values

$$\begin{aligned}
k_{x1} &= \Delta t \cdot \mathbf{f}(\mathbf{x}(t)) \\
k_{x2} &= \Delta t \cdot \mathbf{f}\left(\mathbf{x}(t) + \frac{k_{x1}}{2}\right) \\
k_{x3} &= \Delta t \cdot \mathbf{f}\left(\mathbf{x}(t) + \frac{k_{x2}}{2}\right) \\
k_{x4} &= \Delta t \cdot \mathbf{f}(\mathbf{x}(t) + k_{x3}),
\end{aligned} \tag{5.40}$$

and evolves the state forward using

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \frac{1}{6}(k_{x1} + 2k_{x2} + 2k_{x3} + k_{x4}). \tag{5.41}$$

The simulation of the reservoir dynamics requires more careful analysis, because it is a system driven by external inputs. For the general reservoir dynamics

$$\frac{1}{\gamma} \dot{\mathbf{r}} = \mathbf{f}(\mathbf{r}, \mathbf{x}, \mathbf{c}) = -\mathbf{r} + \mathbf{g}(A\mathbf{r} + B\mathbf{x} + C\mathbf{c} + \mathbf{d}), \quad (5.42)$$

the algorithm to update the reservoir states is given by

$$\begin{aligned} k_{r1} &= \Delta t \cdot \mathbf{f}(\mathbf{r}(t), \mathbf{x}(t), \mathbf{c}(t)) \\ k_{r2} &= \Delta t \cdot \mathbf{f}\left(\mathbf{r}(t) + \frac{k_{r1}}{2}, \mathbf{x}(t) + \frac{k_{x1}}{2}, \mathbf{c}(t) + \frac{k_{c1}}{2}\right) \\ k_{r3} &= \Delta t \cdot \mathbf{f}\left(\mathbf{r}(t) + \frac{k_{r2}}{2}, \mathbf{x}(t) + \frac{k_{x2}}{2}, \mathbf{c}(t) + \frac{k_{c2}}{2}\right) \\ k_{r4} &= \Delta t \cdot \mathbf{f}(\mathbf{r}(t) + k_{r3}, \mathbf{x}(t) + k_{x3}, \mathbf{c}(t) + k_{c3}), \end{aligned} \quad (5.43)$$

and the reservoir state evolves forward according to

$$\mathbf{r}(t + \Delta t) = \mathbf{r}(t) + \frac{1}{6}(k_{r1} + 2k_{r2} + 2k_{r3} + k_{r4}). \quad (5.44)$$

Hence, when we simulate the Lorenz state  $\mathbf{x}(t)$ , we also save the corresponding values  $k_{x1}, \dots, k_{x3}$  to use in the reservoir update algorithm. Finally, we note that in our simulations, we slowly vary the control input  $\mathbf{c}(t)$  over time, requiring us to determine the trajectory of  $\mathbf{c}(t)$  beforehand. However, we require the differential equation that generated  $\mathbf{c}(t)$  to solve for the final parameters  $k_{c1}, \dots, k_{c4}$ . We assume the differential equations that generate  $\mathbf{c}$  are constant, such that between time  $t$  and  $t + \Delta t$ , the rate of change of  $\mathbf{c}(t)$  is given by

$$\dot{\mathbf{c}}(t) = \mathbf{f}(\mathbf{c}(t)) = \frac{\mathbf{c}(t + \Delta t) - \mathbf{c}(t)}{\Delta t}. \quad (5.45)$$

Such dynamics yield the parameters

$$\begin{aligned}
k_{c1} &= \Delta t \cdot \mathbf{f}(\mathbf{c}(t)) = \mathbf{c}(t + \Delta t) - \mathbf{c}(t) \\
k_{c2} &= \Delta t \cdot \mathbf{f}\left(\mathbf{c}(t) + \frac{k_{c1}}{2}\right) = \Delta t \cdot \mathbf{f}\left(\frac{\mathbf{c}(t + \Delta t) + \mathbf{c}(t)}{2}\right) = \mathbf{c}(t + \Delta t) - \mathbf{c}(t) \\
k_{c3} &= \Delta t \cdot \mathbf{f}\left(\mathbf{c}(t) + \frac{k_{c2}}{2}\right) = \Delta t \cdot \mathbf{f}\left(\frac{\mathbf{c}(t + \Delta t) + \mathbf{c}(t)}{2}\right) = \mathbf{c}(t + \Delta t) - \mathbf{c}(t).
\end{aligned} \tag{5.46}$$

The same integration is used with feedback where  $\frac{1}{\gamma}\dot{\mathbf{r}} = \mathbf{f}(\mathbf{r}, \mathbf{c}) = -\mathbf{r} + \mathbf{g}(\mathbf{R}\mathbf{r} + \mathbf{C}\mathbf{c} + \mathbf{d})$ .

### 5.9. Training for tanh

Using the dynamical equations and RK4 integration scheme, we first generated the training inputs  $\mathbf{x}(t)$ . As examples, we consider the single direction translation and transformation examples described in the main text using four Lorenz attractor inputs. The first was the original Lorenz time series  $\mathbf{x}(t)$ , and the remaining three were translations or rotations of the original. Each of these four time series were simulated for  $T = T_{\text{waste}} + T_{\text{train}} = 20 + 200 = 220$  time. At a time step of  $\Delta t = 0.001$ , each time series  $\mathbf{x}(t)$  contained  $\frac{T}{\Delta t} = 220,000$  simulation time points, stored in data matrix  $X_0$  for the original attractor. Because we also kept the 4 outputs of the RK4 numerical integration scheme, the data matrix  $X_0$  had dimensions variables  $\times$  time steps  $\times$  RK4  $= 3 \times 220,000 \times 4$ . With three additional time series for translation or rotation,  $X_1, X_2, X_3$ , we concatenated the four time series along the second dimension into the full matrix  $X$  with dimension  $3 \times 880,000 \times 4$ .

Using this Lorenz data matrix  $X$ , and a corresponding control input data matrix, we drove the reservoir to generate  $\mathbf{r}(t)$ , contained in a reservoir data matrix  $D$  that was of size  $N \times 880,000$ . For every  $\frac{T}{\Delta t} = 220,000$  time steps, we threw away the first  $\frac{T_{\text{waste}}}{\Delta t} = 20,000$  time points, as this simulation allowed both the Lorenz and reservoir systems to forget their initial conditions. The remaining  $\frac{T_{\text{train}}}{\Delta t} = 200,000$  time points of each attractor were kept for training. This process yields a Lorenz training matrix  $X_{\text{train}}$  of dimension  $3 \times 800,000$  (as we throw away the RK4 simulation parameters after driving the reservoir), and a reservoir

training matrix  $D_{\text{train}}$  of dimension  $N \times 800,000$ .

Finally, we seek a training matrix  $W$  of dimension  $3 \times N$  that minimizes the matrix 2-norm

$$\|WD_{\text{train}} - X_{\text{train}}\|_2. \quad (5.47)$$

Specifically, we use MATLAB's command `lsqminnorm`, that not only minimizes this norm, but in the event that multiple solutions exist, also minimizes the norm of  $W$ .

### 5.10. Training for Wilson-Cowan

When training the Wilson-Cowan system, we made a slight modification to preserve some of the biophysically motivated aspects of the connectivity. Recall that while the excitatory populations were allowed to connect to both excitatory and inhibitory populations of any oscillator, the inhibitory populations were only allowed to connect to the excitatory and inhibitory population of their own oscillator. Hence, if we perform feedback using all  $N$  populations (comprising  $N/2$  oscillators), then the inhibitory populations would gain many more connections that were previously not allowed.

Hence, when we perform training and feedback for the WC reservoir, we only use the activity of the excitatory populations to reconstruct the input. Put another way, if the reservoir state is organized such that the excitatory populations  $\mathbf{E}$  are ordered before the inhibitory populations  $\mathbf{I}$  where

$$\mathbf{r}(t) = \begin{bmatrix} \mathbf{E} \\ \mathbf{I} \end{bmatrix}, \quad (5.48)$$

then the matrix  $W$  is only trained on the first  $N/2$  states of  $\mathbf{r}$ . Hence,  $W$ , which has dimensions  $M \times N$ , contains non-zero entries for the first  $N/2$  columns, and zero entries for the second  $N/2$  columns. Then, when we perform feedback such that  $R = A + BW$ , we preserve the original structure whereby inhibitory populations only connect to populations of their own oscillator.

### 5.11. Truncation of the Block-Hessenberg Matrix

To understand the mechanism of learning translations and transformations, we had taken the differential of the reservoir feedback dynamics,

$$(I - KA)d\mathbf{r}' + \frac{1}{\gamma}d\dot{\mathbf{r}}' = K(BWd\mathbf{r}' + Cdc). \quad (5.49)$$

If we take time derivatives of the left-hand side of this equation, we obtain

$$\begin{bmatrix} (I - KA) & \frac{1}{\gamma}I & 0 & 0 & 0 & \dots \\ \downarrow \searrow & & \searrow & & & \\ -\dot{K}A & (I - KA) & \frac{1}{\gamma}I & 0 & 0 & \dots \\ \downarrow \searrow & \downarrow \searrow & & \searrow & & \\ -\ddot{K}A & -2\dot{K}A & (I - KA) & \frac{1}{\gamma}I & 0 & \dots \\ \downarrow \searrow & \downarrow \searrow & \downarrow \searrow & \searrow & & \\ -\dddot{K}A & -3\ddot{K}A & -3\dot{K}A & (I - KA) & \frac{1}{\gamma}I & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} d\mathbf{r}' \\ d\dot{\mathbf{r}}' \\ d\ddot{\mathbf{r}}' \\ d\ddot{\mathbf{r}}' \\ \vdots \end{bmatrix}, \quad (5.50)$$

where the element in the  $i$ -th row and  $j$ -th column has a coefficient

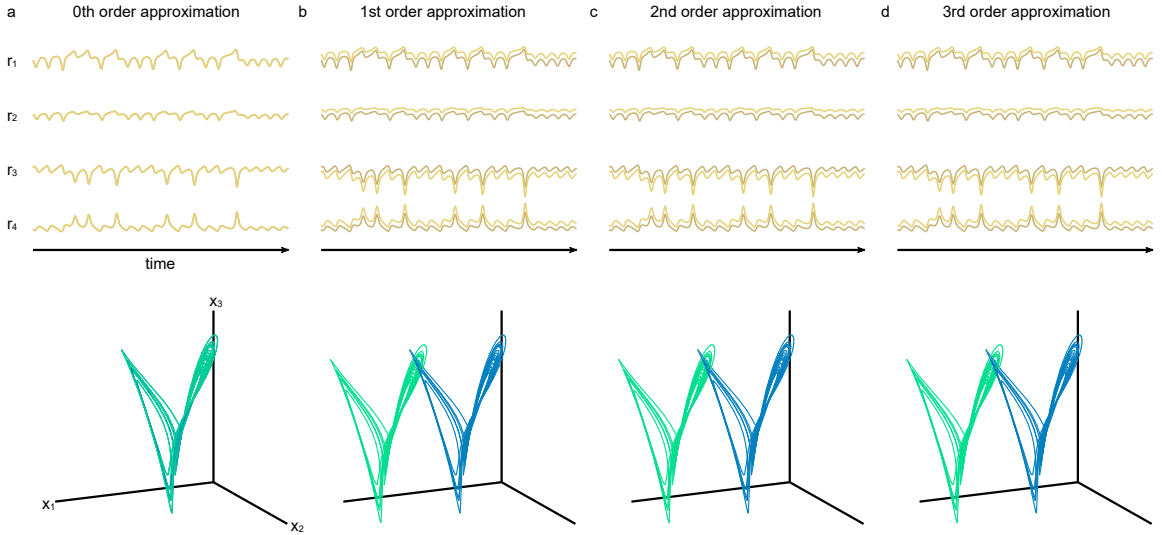
$$p_{i,j} = \binom{i-1}{j-1} \quad \text{for } j \leq i, \quad (5.51)$$



according to Pascal's triangle. For the translation examples, we can write the continued time derivatives of the differential relation as

$$\underbrace{\begin{bmatrix} H_0 & H_{-1} & 0 & 0 & \cdots \\ H_1 & H_0 & H_{-1} & 0 & \cdots \\ H_2 & 2H_1 & H_0 & H_{-1} & \cdots \\ H_3 & 3H_2 & 3H_1 & H_0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}}_J \begin{bmatrix} d\mathbf{r}' \\ d\dot{\mathbf{r}}' \\ d\ddot{\mathbf{r}}' \\ d\dddot{\mathbf{r}}' \\ \vdots \end{bmatrix} \approx \begin{bmatrix} K \\ \dot{K} \\ \ddot{K} \\ \dddot{K} \\ \vdots \end{bmatrix} (BP + C)d\mathbf{c}, \quad (5.52)$$

where  $H_{-1} = \frac{1}{\gamma}I$ ,  $H_0 = I - KA$ , and  $H_i = -K^{(i)}A$  is the  $i$ -th time-derivative of  $KA$ . This matrix is a block matrix (each element  $H$  is a matrix), and is specifically a block-Hessenberg matrix (zero above the first block-super diagonal).



**Figure 30: Predicted Change in Reservoir States Given a Change in Control Parameter.** Reservoir time series generated by driving the reservoir with the original Lorenz input with  $c = 0$  (dark gold), and the predicted time series from solving for  $d\mathbf{r}'$  after training on translated examples and changing the control parameter  $\Delta c = 20$  (light gold), along with their output projections (blue and green, respectively). These approximations were taken by computing the inverse Eq. 5.55 for (a)  $k = 0$ , (b)  $k = 1$ , (c)  $k = 2$ , and (d)  $k = 3$ .

The goal is to solve for  $d\mathbf{r}'$  with respect to  $d\mathbf{c}$ . If we truncate  $J$  to a finite-dimensional

matrix such that

$$J \simeq \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix}, \quad (5.53)$$

where

$$\begin{aligned} J_{11} &= \begin{bmatrix} H_0 \\ H_1 \\ \vdots \\ H_{k-1} \end{bmatrix}, & J_{12} &= \begin{bmatrix} H_{-1}, & 0, & \cdots, & 0 \\ H_0, & H_{-1}, & \cdots, & 0 \\ \vdots & \vdots & \ddots & \vdots \\ p_{k,2}H_{k-2}, & p_{k,3}H_{k-3}, & \cdots, & H_{-1} \end{bmatrix}, \\ J_{21} &= \begin{bmatrix} p_{k+1,1}H_k \end{bmatrix}, & J_{22} &= \begin{bmatrix} p_{k+1,2}H_{k-1} & p_{k+1,3}H_{k-2} & \cdots & H_0 \end{bmatrix}, \end{aligned} \quad (5.54)$$

Then, the closed form solution for the first  $N$  rows of  $J^{-1}$  (the first block) can be written as

$$[J^{-1}]_{(1:N,:)} \simeq -(J_{22}J_{12}^{-1}J_{11} - J_{21})^{-1} \begin{bmatrix} -J_{22}J_{12}^{-1} & I \end{bmatrix}. \quad (5.55)$$

However, in reality,  $J$  is not a finite matrix, but an infinite dimensional matrix. An important fact to verify, then, is whether there exists a sufficiently large value of  $k$  to yield an accurate inversion. While proving that this inverse converges is outside the scope of this work, we numerically demonstrate in what follows that after  $k = 1$ , successive terms do not perceptibly change the results. Specifically, we solve for  $d\mathbf{r}'$  with respect to  $d\mathbf{c}$  for  $k = 0, 1, 2, 3$ .

As a reference for translation, at  $k = 0$ , the approximation becomes

$$d\mathbf{r}' \approx H_0^{-1}K(BP + C)d\mathbf{c}, \quad (5.56)$$

and at  $k = 1$ , we obtain the approximation used in the main text. The 0-th order approximation at  $k = 0$  yields no change (Fig. 30a), where the predicted reservoir states (light gold) are identical to the original states (dark gold). The first order approximation at  $k = 1$  (Fig. 30b) yields a change in the reservoir states that outputs to the expected translation

in spatial coordinates. Taking more terms in the approximation ( $k = 2$ , Fig 30c; and  $k = 3$ , Fig. 30d) yields no perceivable change in either the reservoir states or their outputs.

### 5.12. Simulation of the Jansen Linkage

Because the Jansen linkage is a kinematic system, it requires several modifications to simulate its motion. We begin by considering two nodes  $i$  and  $j$  that are connected by a link  $k$  of fixed squared length  $l = l_k^2$ , and located at  $\mathbf{x}_i$  and  $\mathbf{x}_j$ , respectively. The link fixes the distance between the two nodes, such that

$$(\mathbf{x}_i - \mathbf{x}_j)^\top (\mathbf{x}_i - \mathbf{x}_j) = l. \quad (5.57)$$

The allowed infinitesimal motions are given by taking the total derivative of the constraint

$$(\mathbf{x}_i - \mathbf{x}_j)^\top (d\mathbf{x}_i - d\mathbf{x}_j) = 0. \quad (5.58)$$

Now we consider all  $N$  nodes of the linkage and compile their positions into a vector,  $\mathbf{x} = [\mathbf{x}_1; \mathbf{x}_2; \dots; \mathbf{x}_N]$ , and similarly consider all  $E$  links and compile the distance constraints into a vector  $\mathbf{l}(\mathbf{x})$ . By taking the gradient of  $\mathbf{l}(\mathbf{x})$  with respect to  $\mathbf{x}$  at a specific position  $\mathbf{x}^*$ , we obtain all allowed infinitesimal motions of the linkage

$$\nabla_{\mathbf{x}} \mathbf{l}|_{\mathbf{x}=\mathbf{x}^*} = R(\mathbf{x}^*) d\mathbf{x}, \quad (5.59)$$

where  $R(\mathbf{x})$  is called the *rigidity matrix*, and has dimensions  $E \times 2N$ .

To simulate the motion, we must first ensure that there is no change in link lengths. Hence, the nodes must move in a way that obeys

$$R(\mathbf{x}^*) d\mathbf{x} = \mathbf{0}. \quad (5.60)$$

We notice that because  $\mathbf{x}^*$  is already a fixed constant,  $R(\mathbf{x}^*)$  is also a matrix of constants.

Therefore, the equation is linear in the variables  $d\mathbf{x}$ , and all motions reside in the nullspace

$$d\mathbf{x} \in \mathcal{N}(R(\mathbf{x}^*)). \quad (5.61)$$

The Jansen linkage does not contain any kinematic singularities, where the row-rank of  $R(\mathbf{x}^*)$  loses rank. Hence, the dimension of the nullspace is given simply by  $2N - E = 4$ . Among these four directions are three rigid body motions: the  $x$ -translation,  $y$ -translation, and rotation. In our simulations, we quotient out the rigid body motions, and evolve forward the node positions along the direction of the remaining motion using a step size of 0.05. To ensure numerical accuracy, we evolve forward these kinematics using a 4-th order Runge-Kutta approximation.

### 5.13. Quantifying Prediction Accuracy of Lorenz Computations

In the text, we visually confirm that the reservoir was able to translate, transform, and bifurcate the Lorenz example. To quantify how close the predicted manipulation is to the target manipulation, we numerically compute the shortest Euclidean distance between every point in the predicted time series and the true Lorenz. We begin by simulating a Lorenz time series for  $T_{\text{waste}} = 20$  at  $\Delta t = 0.001$  as an transient period that we throw away, then simulate the Lorenz for  $T = 40,000$ , or 40 million time points, as the reference attractor point set, which we collect into a  $3 \times \frac{T}{\Delta t}$  matrix  $Y$ .

Next, we take the predicted reservoir output, and undo the target manipulation. Specifically, in the translation, changing  $c$  by 1 should translate the reservoir's representation by  $P = [1; 0; 0]$  along the  $x_1$  direction, according to

$$\mathbf{x}_c(t) = \mathbf{x}_0(t) + Pc. \quad (5.62)$$

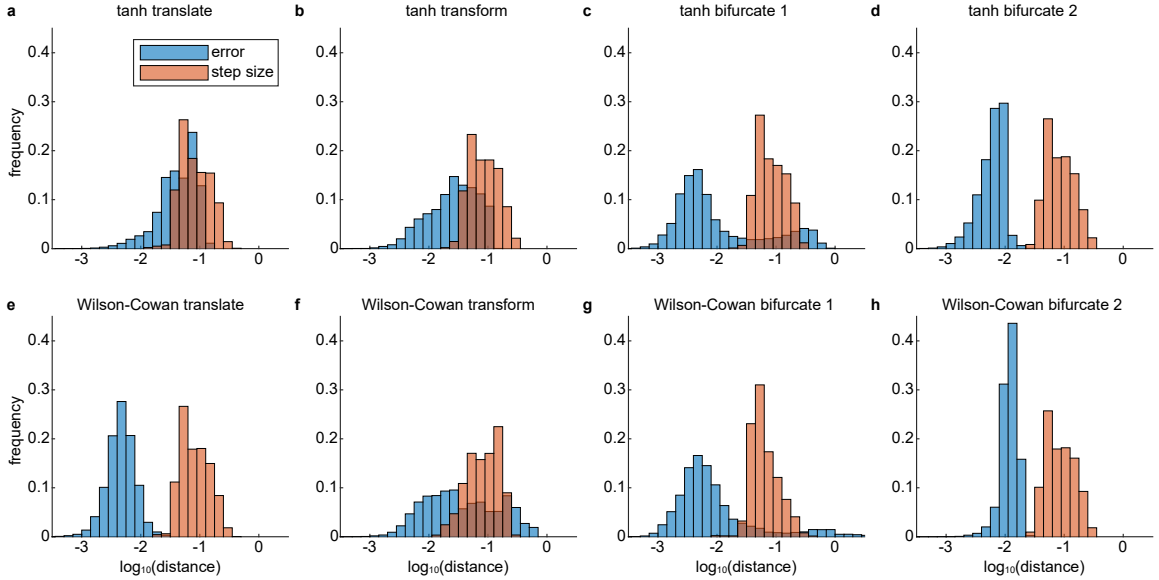
Hence, we simulate the reservoir at  $c = -40$  and  $c = 40$ , thereby generating reservoir time series  $\mathbf{r}_{-40}(t)$  and  $\mathbf{r}_{40}(t)$ , which project to  $\hat{\mathbf{x}}_{-40}(t) = W\mathbf{r}_{-40}(t)$ , and  $\hat{\mathbf{x}}_{40}(t) = W\mathbf{r}_{40}(t)$ .

Then, we perform the inverse manipulation as  $\hat{\mathbf{x}}_{-40}(t) + 40P$  and  $\hat{\mathbf{x}}_{40}(t) - 40P$ .

Similarly, for the transformation, we multiply the time series by  $I + cP$ , where

$$\mathbf{x}_c(t) = (I + Pc)\mathbf{x}_0(t), \quad P = \begin{bmatrix} -0.012 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \quad (5.63)$$

Hence, we simulate the reservoir at  $c = -40$  and  $c = 40$ , thereby generating the reservoir time series and projections. Then, we perform the inverse manipulation as  $(I - 40P)^{-1}\hat{\mathbf{x}}_{-40}(t)$  and  $(I + 40P)^{-1}\hat{\mathbf{x}}_{40}(t)$ . For the bifurcation, because the manifold was not expected to be manipulated, no inverse is performed.



**Figure 31: Tanh & Wilson-Cowan Attractor Similarity.** The distributions of the shortest Euclidean distance between all predicted attractor points (after performing the inverse manipulation) and the reference Lorenz attractor set (blue), as well as of the Euclidean distance of one simulation step for (a) the tanh reservoir undergoing translation in the  $x_1$  direction, (b) the tanh reservoir undergoing a squeeze in the  $x_1$  direction, (c) the tanh reservoir undergoing bifurcation after training on 4 values of  $\rho = 21, 22, 23$ , and 24 at one wing, and (d) the tanh reservoir undergoing bifurcation after training on 2 values of  $\rho = 23$  and 24 at both wings. (e–h) The same distributions for the respective manipulations for the Wilson-Cowan oscillator networks. Each histogram contains the distribution of distances for 50 random instantiations of reservoir and Lorenz parameters.

Finally, we take each of the inverse manipulation time series, and compute the shortest Euclidean distance of each time point to the reference attractor point set  $Y$ . We perform this comparison across 50 random instantiations of reservoir and Lorenz parameters for each manipulation (i.e. translation, transformation, bifurcation) and reservoir type (i.e. tanh, Wilson-Cowan). Then, we plot the distribution of distances for all tanh and Wilson-Cowan examples (Fig. 31).

#### 5.14. Translations, Transformations, & Bifurcations with Wilson-Cowan Networks

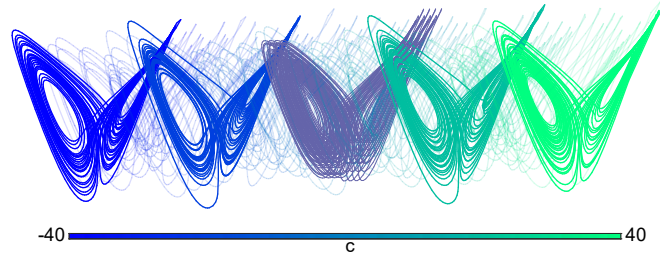


Figure 32: **Translation of the Lorenz Representation Using Wilson-Cowan Oscillator Networks.** Output of the feedback reservoir after being trained on 4 time series of a Lorenz attractor translated in the  $x_1$  direction at  $c = 0, \dots, 3$ . By varying  $c$  from  $-40$  to  $40$ , the representation shifts in the  $x_1$  direction.

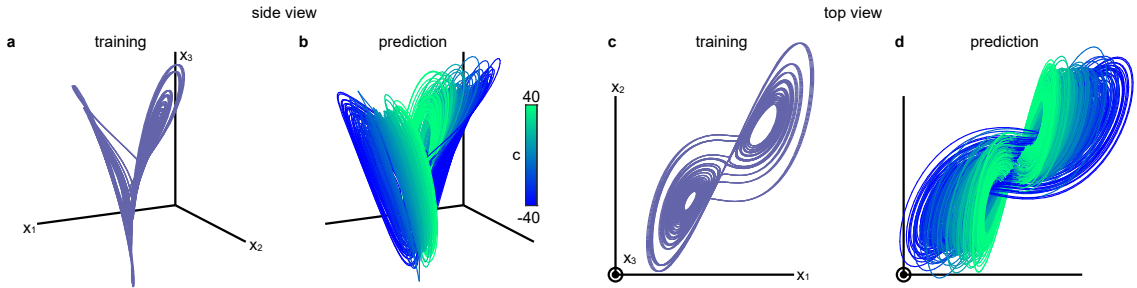


Figure 33: **Transformation of the Lorenz Representation Using Wilson-Cowan Oscillator Networks.** (a,c) Training examples and (b,d) predicted output of the feedback reservoir after being trained on 4 time series of a Lorenz attractor squeezed in the  $x_1$  direction at  $c = 0, \dots, 3$ . By varying  $c$  from  $-40$  to  $40$ , the representation squeezes in the  $x_1$  direction.

Here, we replicate the results of the main text for the translation (Fig. 32), transformation (Fig. 33), and bifurcation inference (Fig. 34) of the Lorenz system at  $\rho \approx 28$  using the

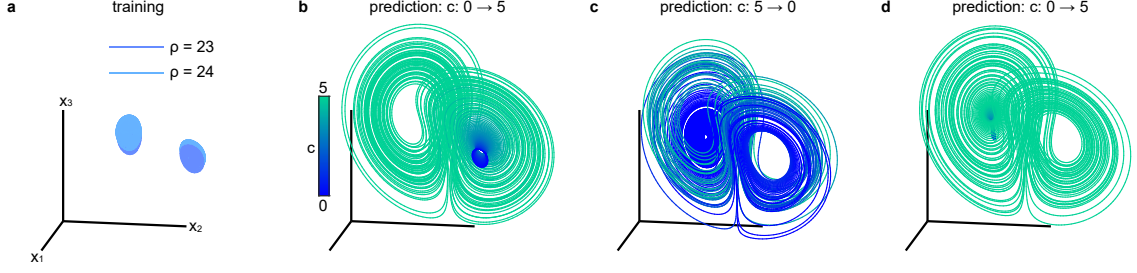


Figure 34: **Bifurcation of the Lorenz Representation Using Wilson-Cowan Oscillator Networks.** (a) Training examples of a pre-bifurcated Lorenz attractor, where the Wilson-Cowan reservoir was trained at  $c = 0$  for  $\rho = 23$ , and  $c = 1$  for  $\rho = 24$ . (b) Output of the feedback reservoir. By varying  $c$  from 0 to 5, the representation correctly bifurcates into the Lorenz-shaped manifold. (b) By varying  $c$  back to 0, the reservoir representation displays transient chaos. (c) By varying  $c$  back to 5, the fixed points bifurcate again to become unstable.

Wilson-Cowan oscillator.

### 5.15. Translation in Multiple Directions

In the main text, we demonstrated that a reservoir can translate its representation of a Lorenz attractor along the  $x_1$  direction. Specifically, we took an untranslated Lorenz time series  $\mathbf{x}_0(t)$ , and generated three additional training examples  $\mathbf{x}_c(t)$  for  $c = 1, 2, 3$  such that

$$\mathbf{x}_c(t) = \mathbf{x}_0(t) + c \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}. \quad (5.64)$$

We then drove the reservoir using these four training examples and an additional control parameter  $c$  that we also varied from  $c = 0, \dots, 4$ . Afterwards, we performed the feedback, and translated the reservoir's representation by varying the external control parameter  $c$  from  $-40$  to  $40$ . We reproduce this translated representation here (Fig. 35a). We show the same output of the feedback reservoir trained on four examples translated in the  $x_2$  direction ( $\mathbf{x}_c(t) = \mathbf{x}_0(t) + c[0; 1; 0]$ ) and in the  $x_3$  direction ( $\mathbf{x}_c(t) = \mathbf{x}_0(t) + c[0; 0; 1]$ ) (Fig. 35b,c). Hence, we demonstrate that the reservoir can learn these translations in arbitrary directions.

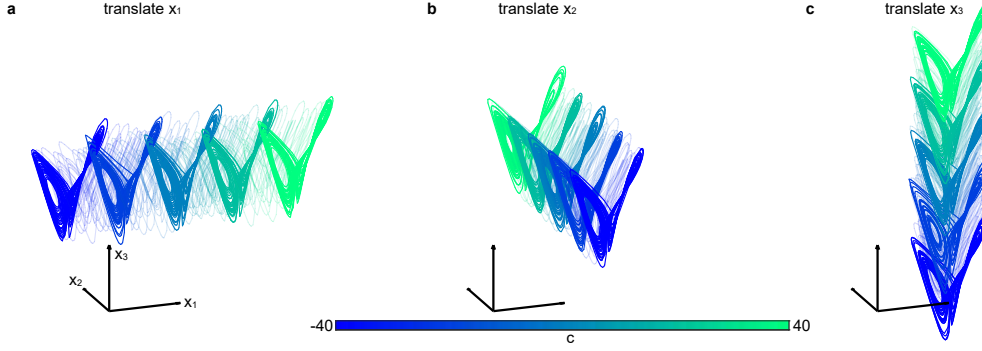


Figure 35: **Translation of the Lorenz Representation in all Three Spatial Directions.** (a) Output of the feedback reservoir after being trained on 4 time series of a Lorenz attractor translated in the  $x_1$  direction at  $c = 0, \dots, 4$ . By varying  $c$  from  $-40$  to  $40$ , the representation shifts in the  $x_1$  direction. (b) The same scheme is employed for translations in the  $x_2$  direction, and (c) in the  $x_3$  direction.

### 5.16. Different Types of Transformations

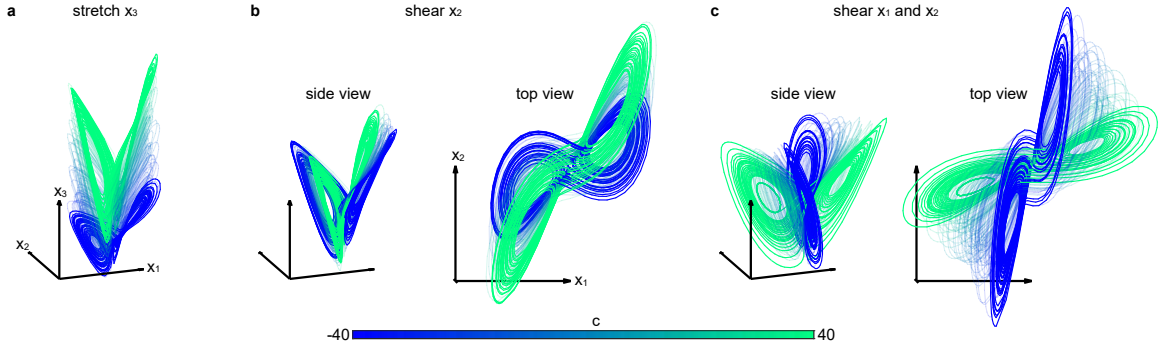
In the main text, we demonstrated that a reservoir trained on the original Lorenz attractor  $\mathbf{x}_0(t)$  and on three transformed examples  $\mathbf{x}_c(t) = (I + cP)\mathbf{x}_0(t)$  for  $c = 1, 2, 3$ , was able to continuously interpolate and extrapolate the transformation on its internal representation, even for control inputs between  $-40$  and  $40$ . Here, we consider a stretch in the  $x_3$  direction, a shear in the  $x_1$  direction, and a shear in the  $x_1$  and  $x_2$  directions. Specifically, we use the



three matrices

$$\begin{aligned}
P_{\text{stretch},x_3} &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0.012 \end{bmatrix}, \\
P_{\text{shear},x_1} &= \begin{bmatrix} 0 & 0 & 0 \\ 0.012 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \\
P_{\text{shear},x_1,x_2} &= \begin{bmatrix} 0 & -0.012 & 0 \\ 0.012 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},
\end{aligned} \tag{5.65}$$

to train our reservoir for  $c = 0, \dots, 4$ . For each transformation, we drive the reservoir with the input Lorenz attractors  $\mathbf{x}_c(t)$  and an additional control input  $c$  for  $c = 0, \dots, 4$ . We then train the reservoir by applying the feedback method used in the main text. Finally, we drive the autonomous feedback reservoir by varying the control parameter from  $c = -40$  to  $c = 40$  for these three transformations (Fig. 36).



**Figure 36: Transformation of the Lorenz Representation Using Stretch & Shear in Several Spatial Directions.** (a) Output of the feedback reservoir after being trained on 4 time series of a Lorenz attractor stretched in the  $x_3$  direction at  $c = 0, \dots, 4$ . By varying  $c$  from  $-40$  to  $40$ , the representation stretches in the  $x_3$  direction. (b) The same scheme is employed for a shear in the  $x_2$  direction, and (c) for a shear in the  $x_1$  and  $x_2$  directions.

## CHAPTER 6 : Conformational Control of Mechanical Networks

### 6.1. Motivation

Many physical systems can be thought of as networks in which contacts, bonds, linkages, or hinges connect physical elements to one another. From the study of force chains in granular materials (Papadopoulos et al., 2018) to the study of fiber networks in polymer physics (Picu, 2011), it has become clear that both regular and irregular patterns of connectivity between physical elements constrain the bulk properties of the material, including its response to stress and shear (Vermeulen et al., 2017), its ability to transmit acoustic signals (Bassett et al., 2012), and its capacity for thermal and electrical transport (Shi et al., 2014). These networks are also integral to the ever-evolving exploration of everyday machines in robotics (Detweiler et al., 2007) and biology (Patek et al., 2007b). Perhaps one of the simplest and most powerful conceptual advances in understanding such systems was the development of structural rigidity theory (Crapo, 1979b), built on a seminal early paper on constraint counting from J.C. Maxwell (Maxwell, 1864a), in which one predicts the flexibility of ensembles formed by rigid bodies connected by flexible linkages (Grimm and Dorner, 1975). Frames – consisting of rigid elements (*sites*) and the connections between them (*bonds*) – are said to be rigid when the distance between two points can only be altered by changing the length of at least one connection.

Notably, even in rigid frames, mechanical networks can undergo conformational changes that drastically alter their function, such as exotic shape transformations in metamaterials (Bertoldi et al., 2017a), and allosteric regulation of enzymes where substrate binding in one region changes the structure and function of a distal active site (Guo and Zhou, 2016). Characterizing and subsequently controlling such changes is of critical import to a theoretical understanding of these systems, which in turn will support their novel design and use. Yet, such characterization and control is challenged by the fact that perturbation to a few regions in the network can lead to complex, wide-scale changes in the material’s form that has to-date eluded formal treatment. Some have sought to address this challenge by design-

ing networks through *kinematic synthesis*, tracing arbitrary trajectories with a trace point using only a few actuators (Kempe, 1875). Others have used computational heuristics such as tuning-by-pruning to predict mechanical responses in multiple nodes (Goodrich et al., 2015). Given that such heuristics exist, it is now natural and timely to build a simple theory for how a mechanical network's topology constrains its control, and how novel topologies can be constructed to produce specified control functions. Here we develop and exercise such a simple theory by building on prior work on the deformation of general and bipartite frames (Bolker and Roth, 1980).

## 6.2. Network Connectivity & Mathematical Framework

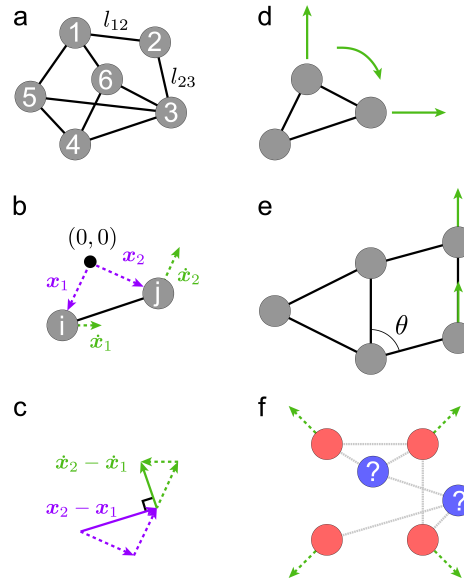


Figure 37: **Graphical Representations of Maxwell Frames.** (a) An example of a rigid frame in  $d = 2$  dimensions with  $N = 6$  nodes and  $E = 9$  edges, marked with the length of the edge connecting node 1 to node 2, and the length of the edge connecting node 2 to node 3. (b) Two nodes connected by one edge, with the position vectors from an arbitrary origin specified in purple, and allowed motion vectors in green, with (c) a graphical representation of the orthogonality of position and motion vectors satisfying Eq. 6.1. (d) Graph of a rigid three-node system with the three rigid body translation and rotation motions in green. (e) Graph of a non-rigid five-node system with the fourth non-rigid body motion shown with green arrows, and parameterized by the continuous variable  $\theta$ . (f) Graph of four red specified nodes with desired motion  $\dot{x}_S$  shown with green arrows, potential edges in gray dashed lines, and unspecified nodes in blue.

Consider a set of nodes  $\mathcal{V} = \{1, \dots, N\}$  in  $d$ -dimensional space, where any node  $i$  has position specified by column vector  $\mathbf{x}_i(t) \in \mathbb{R}^d$  at some time  $t \geq 0$ . Further, consider a set of  $E$  edges  $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ , where each edge  $k$  connecting node pair  $(i, j)$  has a squared length proportional to  $g_k = \frac{1}{2}(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j)$  (Fig. 37a). To enforce rigid edges, we require that the length remains constant in time (Fig. 37b) by setting the time derivative equal to 0,

$$\dot{g}_k = (\mathbf{x}_i - \mathbf{x}_j)^T(\dot{\mathbf{x}}_i - \dot{\mathbf{x}}_j) = 0, \quad (6.1)$$

where the infinitesimal motion  $\dot{\mathbf{x}}_i - \dot{\mathbf{x}}_j$  is perpendicular to the edge  $\mathbf{x}_i - \mathbf{x}_j$ . (Fig. 37c). For convenience, we collect all node positions into column vector  $\mathbf{x} = [\mathbf{x}_1; \dots; \mathbf{x}_N] \in \mathbb{R}^{dN}$ , and all edge lengths into column vector  $\mathbf{g}(\mathbf{x}) = [g_1; \dots; g_E] \in \mathbb{R}^E$ . Because the motions are linear with respect to the positions, we can write the constraints Eq. 6.1 in matrix form

$$\dot{\mathbf{g}}(\mathbf{x}, \dot{\mathbf{x}}) = R(\mathbf{x})\dot{\mathbf{x}} = \mathbf{0}, \quad (6.2)$$

where the  $k$ -th row of the *rigidity matrix*  $R = R(\mathbf{x})$  has all zero entries except  $(\mathbf{x}_i - \mathbf{x}_j)^T$  that multiplies  $\dot{\mathbf{x}}_i$ , and  $(\mathbf{x}_j - \mathbf{x}_i)^T$  that multiplies  $\dot{\mathbf{x}}_j$ . Because Eq. 6.2 is linear in the motions, the motions satisfying the equation reside in the nullspace of the rigidity matrix  $\dot{\mathbf{x}} \in \mathcal{N}(R)$ .

Among these motions are  $d(d+1)/2$  finitely movable rigid body translations and rotations that preserve distances between all nodes. In addition, we define a *conformational motion* to be a non-rigid body motion that satisfies the instantaneous distance constraints Eq. 6.2. Provided that there are no states of self stress (Guest, 2006) such that  $R$  has full row rank, these conformational motions are finitely deformable (see Methods), and the number of finite conformational motions  $D$  is given (Asimow and Roth, 1978) by the number of state

variables subtracted by the number of constraints and rigid body motions

$$D = d \cdot N - E - \frac{d(d+1)}{2}.$$

As a simple example, consider a triangle (Fig. 37d) in  $d = 2$  dimensions with  $N = 3$  nodes and  $E = 3$  edges, such that  $D = 0$ . The only motions are rigid body  $x$ -translation,  $y$ -translation, and rotation, and the frame's configuration is fully determined by fixing 3 non-redundant  $x$  or  $y$  coordinates. Next we consider a more complex network of  $N = 5$  nodes and  $E = 6$  edges such that  $D = 1$  (Fig. 37e). This conformational motion is parameterized by  $\theta$ , which requires the setting of an additional fourth coordinate.

In pursuing the understanding and control of mechanical materials, we are often interested in both the positions and motions  $\mathbf{x}_S, \dot{\mathbf{x}}_S \in \mathbb{R}^{dn}$  in a subset of  $n$  nodes (which we call the *specified nodes*)  $\mathcal{V}_S \subset \mathcal{V}$ , but not those  $\mathbf{x}_U, \dot{\mathbf{x}}_U \in \mathbb{R}^{dm}$ , of the remaining  $m$  nodes (which we call the *unspecified nodes*)  $\mathcal{V}_U \subset \mathcal{V}$ . Considering a subset of specified nodes is common in the study of several materials, such as those that have a negative Poisson ratio (Fig. 37f). In what follows, we demonstrate how these principles can be used to design networks that finitely generate these desired motions by controlling only a few nodes.

### 6.3. Conic Sections & Overlaps of Bipartite Networks

We begin by considering bipartite frames with only edges between all specified and unspecified nodes such that  $\mathcal{E} = \mathcal{V}_S \times \mathcal{V}_U$  (Fig. 38a). We then fix the positions and motions  $\mathbf{x}_S, \dot{\mathbf{x}}_S$  of the specified nodes as constants, and solve for all  $\mathbf{x}_{Uj}, \dot{\mathbf{x}}_{Uj}$  of the unspecified node  $j$  that satisfy the edge constraints Eq. 6.2. In doing so, we retain the specified motions in the conformational motion. As examples in  $d = 2$ , we show one position (blue node) and motion (blue arrow) of an unspecified node that satisfies edge constraints connected to two (Fig. 38b) and three (Fig. 38c) specified nodes (red nodes). We will refer to the *solution space*  $\mathcal{M} \subseteq \mathbb{R}^d$  as the set of all unspecified node positions  $\mathbf{x}_{Uj} \in \mathcal{M}$  that satisfy constraints

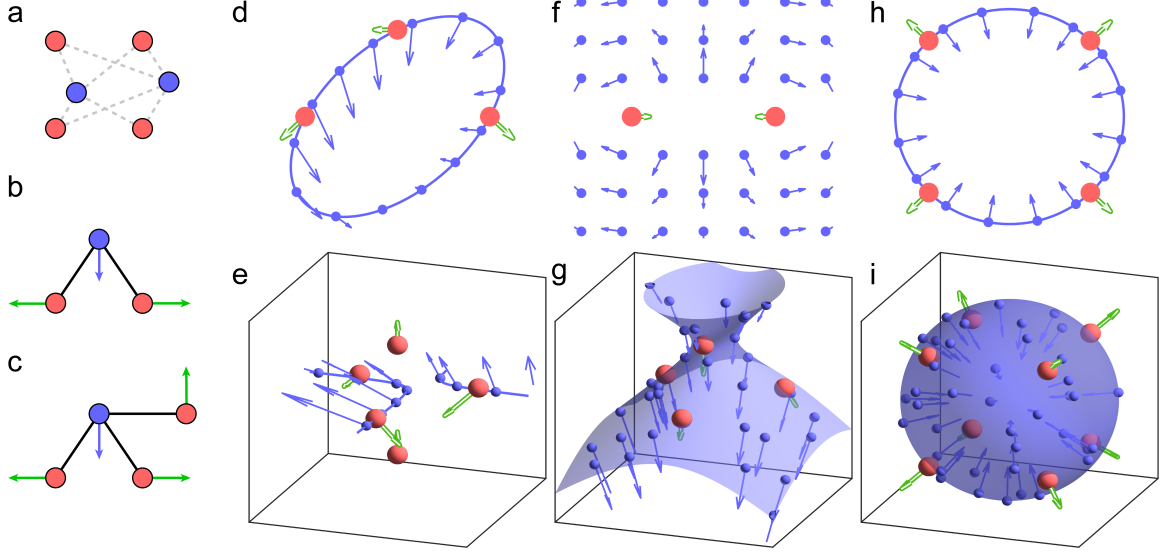


Figure 38: **Solution Space of Unspecified Nodes is Determined by the Specified Nodes.** (a) Example of a bipartite network with specified nodes shown in red, unspecified nodes shown in blue, and allowed edges shown in gray. (b) The position ( $\mathbf{x}_{Uj}$ , location of blue node) and motion ( $\dot{\mathbf{x}}_{Uj}$ , blue arrow) of an unspecified node  $j$  connected to two specified nodes (red), and (c) three specified nodes (also red), with motion  $\dot{\mathbf{x}}_S$  shown with green arrows. In both cases, the blue node and the blue arrow represent one position  $\mathbf{x}_{Uj}$  and one motion  $\dot{\mathbf{x}}_{Uj}$  satisfying Eq. 6.1. (d) One dimensional solution spaces  $\mathcal{M}$  of all possible positions (blue curve) and motion (blue arrows) of an unspecified node (in blue) connected to all specified nodes (in red) with specified motions (hollow green arrows) for  $d = 2$ , and (e) for  $d = 3$ . (f) Two dimensional solution spaces  $\mathcal{M}$  in  $d = 2$ , and (g) in  $d = 3$ . (h) Solution spaces where the specified node positions and motions are redundant to yield a larger than expected solution space for  $d = 2$ , and (i) for  $d = 3$ .

(Eq. 6.2).

We begin solving for solution space  $\mathcal{M}$  by writing the  $n$  edge constraints from Eq. 6.2

$$\begin{bmatrix} (\mathbf{x}_{S1} - \mathbf{x}_{Uj})^T (\dot{\mathbf{x}}_{S1} - \dot{\mathbf{x}}_{Uj}) \\ \vdots \\ (\mathbf{x}_{Sn} - \mathbf{x}_{Uj})^T (\dot{\mathbf{x}}_{Sn} - \dot{\mathbf{x}}_{Uj}) \end{bmatrix} = \mathbf{0},$$

and we rewrite them by treating the unspecified node positions and motions as variables  $\mathbf{v}$

$$\underbrace{\begin{bmatrix} \dot{\mathbf{x}}_{S1}^T & \mathbf{x}_{S1}^T & -1 \\ \vdots & \vdots & \vdots \\ \dot{\mathbf{x}}_{Sn}^T & \mathbf{x}_{Sn}^T & -1 \end{bmatrix}}_M \underbrace{\begin{bmatrix} \mathbf{x}_{Uj} \\ \dot{\mathbf{x}}_{Uj} \\ c \end{bmatrix}}_{\mathbf{v}} = \underbrace{\begin{bmatrix} \mathbf{x}_{S1}^T \dot{\mathbf{x}}_{S1} \\ \vdots \\ \mathbf{x}_{Sn}^T \dot{\mathbf{x}}_{Sn} \end{bmatrix}}_{\mathbf{b}}, \quad (6.3)$$

where  $c = \mathbf{x}_{Uj}^T \dot{\mathbf{x}}_{Uj}$  (see Methods). By temporarily omitting this nonlinearity in  $c$ , the system is linear in the variables  $\mathbf{v}$  according to  $M\mathbf{v} = \mathbf{b}$ , with all solutions to  $\mathbf{v}$  as

$$\mathbf{v} = W\boldsymbol{\alpha} + \mathbf{v}^*. \quad (6.4)$$

Here,  $W$  is a matrix with  $k$  columns as linearly independent vectors in the null space  $\mathcal{N}(M)$ , and vector  $\mathbf{v}^*$  is a particular solution provided that one exists in the column space  $\mathbf{b} \in \mathcal{C}(M)$ . Finally, we reincorporate the nonlinearity  $c = \mathbf{x}_{Uj}^T \dot{\mathbf{x}}_{Uj}$  (see Methods) to yield a quadratic boundary condition on  $\boldsymbol{\alpha}$

$$\begin{bmatrix} \boldsymbol{\alpha}^T & 1 \end{bmatrix} \begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ 1 \end{bmatrix} = 0. \quad (6.5)$$

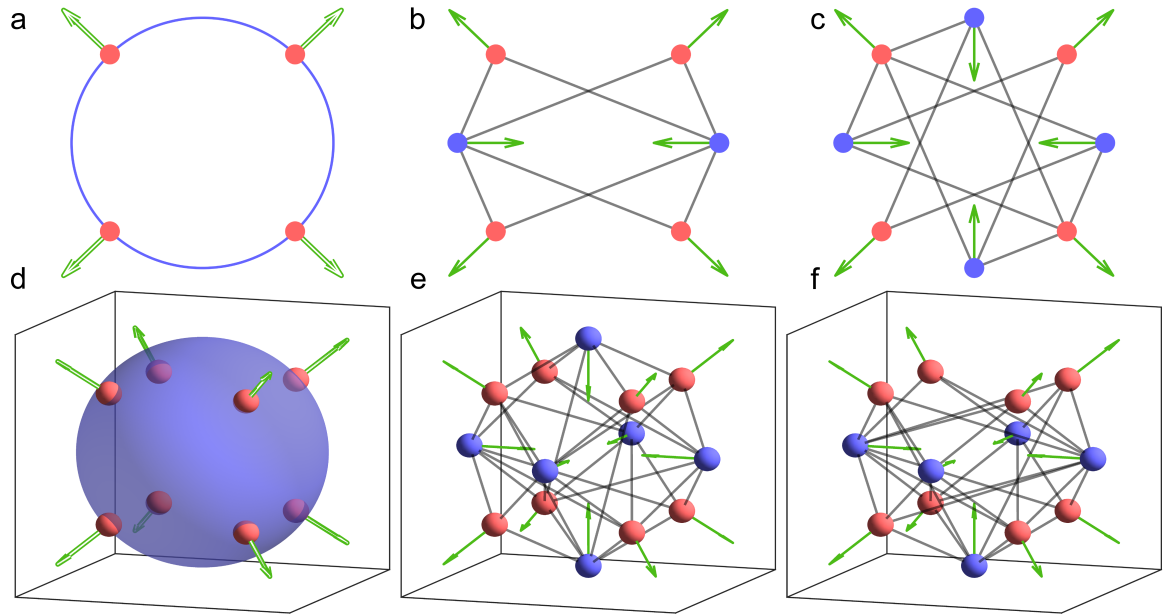
Due to the constraint Eq. 6.5, the solution space has dimension  $\dim(\mathcal{M}) = k - 1$ , which are *points* for  $k = 1$ , *conic sections* for  $k = 2$ , and *quadric surfaces* for  $k = 3$ .

For the general case where the specified node positions and motions are independent ( $M$  has full row rank), the solution space has dimension  $\dim(\mathcal{M}) = 2d - n$ , equal to the number of variables subtracted by the number of constraints. For example, we obtain  $\dim(\mathcal{M}) = 1$  in  $d = 2$  with  $n = 3$  specified nodes (Fig. 38d), and also in  $d = 3$  with  $n = 5$  specified nodes (Fig. 38e). By removing a specified node, we remove an edge constraint and increase the solution space dimension to 2 (Fig. 38f,g).

We can also increase the solution space dimension by creating redundancies in the specified

node positions and motions such that the rows of  $M$  are linearly dependent. For example, in Fig. 38h, the fourth row  $\mathbf{m}_4$  of  $M$  corresponding to the motions and positions of the bottom left node can be written as linear combinations of the first three rows  $\mathbf{m}_1 = [1, 1, 1, 1, -1]$  (top right),  $\mathbf{m}_2 = [-1, 1, -1, 1, -1]$  (top left), and  $\mathbf{m}_3 = [1, -1, 1, -1, -1]$  (bottom right), such that  $\mathbf{m}_4 = \mathbf{m}_2 + \mathbf{m}_3 - \mathbf{m}_1$  in  $d = 2$ , and similarly in  $d = 3$  (Fig. 38i). In summary, these curves and surfaces characterize the only positions  $\mathbf{x}_{Uj}$  of an unspecified node connected to all specified nodes that retain the specified conformational motions. We extend these principles to incorporate connections between unspecified nodes in the Appendix in Chapter 7.

#### 6.4. Network Design Through Judicious Constraint Placement



**Figure 39: Construction & Control of Frames with Specified Outward Motion.** (a) Schematic in  $d = 2$  of four specified nodes with desired outward motion (hollow green arrows) and the corresponding solution space of unspecified node positions (blue curve) satisfying Eq. 6.4. (b) Example bipartite frames with  $D = 1$  finite conformational motion (solid green arrows), constructed from placing 2 unspecified nodes, 8 edges, and (c) 4 unspecified nodes, 12 edges. (d) Schematic in  $d = 3$  of eight specified nodes with desired outward motion (hollow green arrows), with a spherical unspecified node solution space (blue surface). (e) Bipartite frame with  $D = 1$  finite conformational motion, constructed by placing 5 unspecified nodes, 32 edges, and (f) 6 unspecified nodes, 35 edges.



Here we detail how to judiciously constrain a system of  $n$  specified nodes until the desired motion  $\dot{\mathbf{x}}_S$  is the only finite conformational motion. By adding one unspecified node along  $\mathcal{M}$  connected to all specified nodes, we add  $d$  state variables and  $n$  constraints while retaining  $\dot{\mathbf{x}}_S$  as a conformational motion. Similarly, adding  $m$  unspecified nodes along  $\mathcal{M}$  with  $E$  independent edges constrains our system to have  $D$  conformational motions given by Eq. 6.6

$$D = d(n + m) - E - \frac{d(d + 1)}{2}. \quad (6.6)$$

We define our process of *judicious constraint placement* as follows. First, we specify the position  $\mathbf{x}_S$  and motion  $\dot{\mathbf{x}}_S$  of  $n > d$  specified nodes such that they do not all lie on a line in  $d = 2$  or on a plane in  $d = 3$ , and we compute the solution space  $\mathcal{M}$ . Next, we place unspecified nodes  $\mathbf{x}_{Uj} \in \mathcal{M}$  and edges such that (i) we have  $D = 1$  conformational degree of freedom, (ii) there are no states of self stress, and (iii) there are no rigid subgraphs. These conditions ensure that  $\dot{\mathbf{x}}_S$  is in the only finitely deformable conformational motion (see the Appendix in Chapter 7).

As an example in  $d = 2$ , we specify an outward motion in an  $n = 4$  node system (Fig. 39a) with a 1-dimensional solution space. By adding  $m = 2$  nodes with  $E = 8$  edges (Fig. 39b), and  $m = 4$  nodes with  $E = 12$  edges (Fig. 39c) along  $\mathcal{M}$ , we achieve  $\dot{\mathbf{x}}_S$  as the only finite conformational motion with  $D = 1$ . We demonstrate the same result in  $d = 3$  for  $n = 8$  specified nodes (Fig. 39d–f). By judiciously adding unspecified nodes and edges along our solution space, we retain the desired motion  $\dot{\mathbf{x}}_S$  as the sole finite conformational motion.

## 6.5. Multi-Mode Construction & States of Self-Stress

Using this constraint principle, we create networks with two distinct finite conformational motions. Each specified motion  $\dot{\mathbf{x}}_S, \dot{\mathbf{x}}'_S$  generates a solution space  $\mathcal{M}, \mathcal{M}'$  according to Eq. 6.4. By placing our unspecified nodes at their intersection  $\mathbf{x}_{Uj} \in \mathcal{M} \cap \mathcal{M}'$ , we constrain

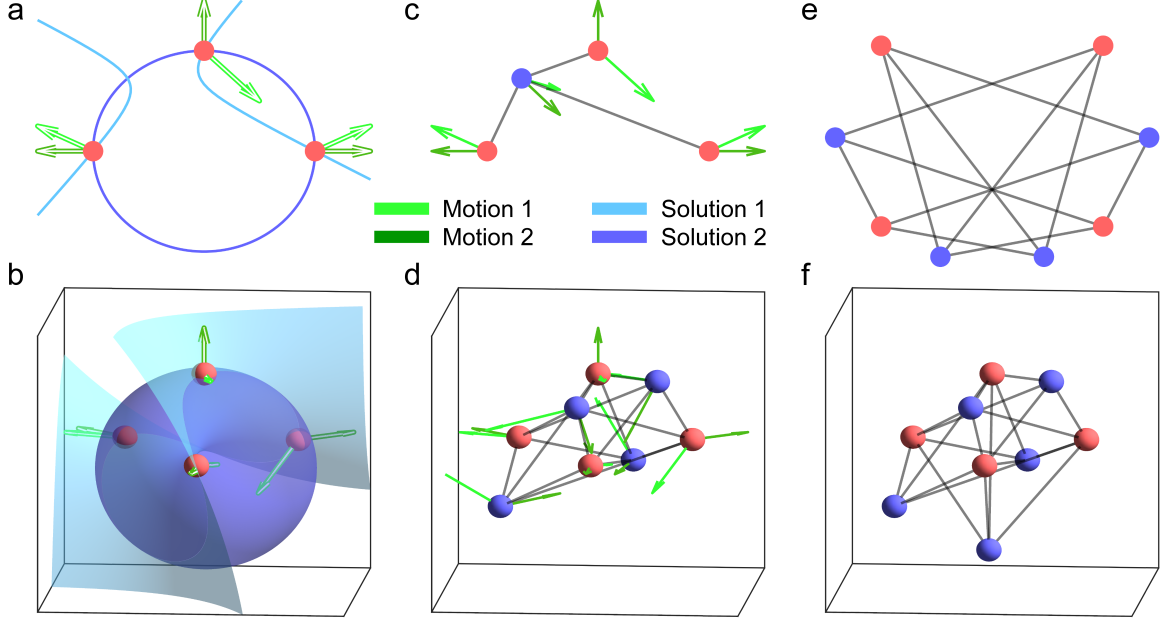


Figure 40: **Intersections of Solution Spaces for Multiple Non-Rigid Motions.** (a) Schematic of two sets of desired motions,  $\dot{\mathbf{x}}_S$  (hollow light green arrows), and  $\dot{\mathbf{x}}'_S$  (hollow dark green arrows) with corresponding solution spaces as light blue and dark blue curves, respectively, for  $n = 3$  specified nodes in  $d = 2$  and (b) for  $n = 4$  specified nodes in  $d = 3$ . (c) The constructed network with the two finite non-rigid body conformational motions shown with solid light and dark green arrows with 1 unspecified node in  $d = 2$ , and (d) with 4 unspecified nodes in  $d = 3$ . (e) The constructed network that should have  $D = 1$  conformational motions by constraint counting, but is actually rigid due to pre-stress stability in  $d = 2$ , and (f) in  $d = 3$ .

the system while allowing both motions. As examples in  $d = 2$  for  $n = 3$  specified nodes (Fig. 40a), and in  $d = 3$  for  $n = 4$  specified nodes (Fig. 40b), we illustrate the solution spaces of two stipulated motions, and place unspecified nodes at their intersection to yield  $D = 2$  conformational motions (Fig. 40cd). Importantly, because  $D = 2$ , we can conform the network along any linear combination of  $\dot{\mathbf{x}}_S, \dot{\mathbf{x}}'_S$ , that requires specifying one extra degree of freedom.

This method can further be used to characterize network geometries that induce self stress. By constraining our network until  $D = 1$  (Eq. 6.6), we expect only one conformational motion in  $\mathcal{N}(R)$ . However, by placing unspecified nodes along the intersection of solution spaces, we allow both desired motions  $\dot{\mathbf{x}}_S, \dot{\mathbf{x}}'_S$  to remain as conformational motions in  $\mathcal{N}(R)$ .

This increase in the nullspace reduces the rank of  $R$  and forces a state of self-stress, allowing us to create networks that are rigid with fewer bonds than required for rigidity, or with branched finite motions (Fig. 40ef, see the Appendix in Chapter 7). Hence, we can generate and avoid networks with self-stress by studying the placement of unspecified nodes at the intersection of solution spaces (see the Appendix in Chapter 7).

## 6.6. Combining Network Modes for Potential Applications

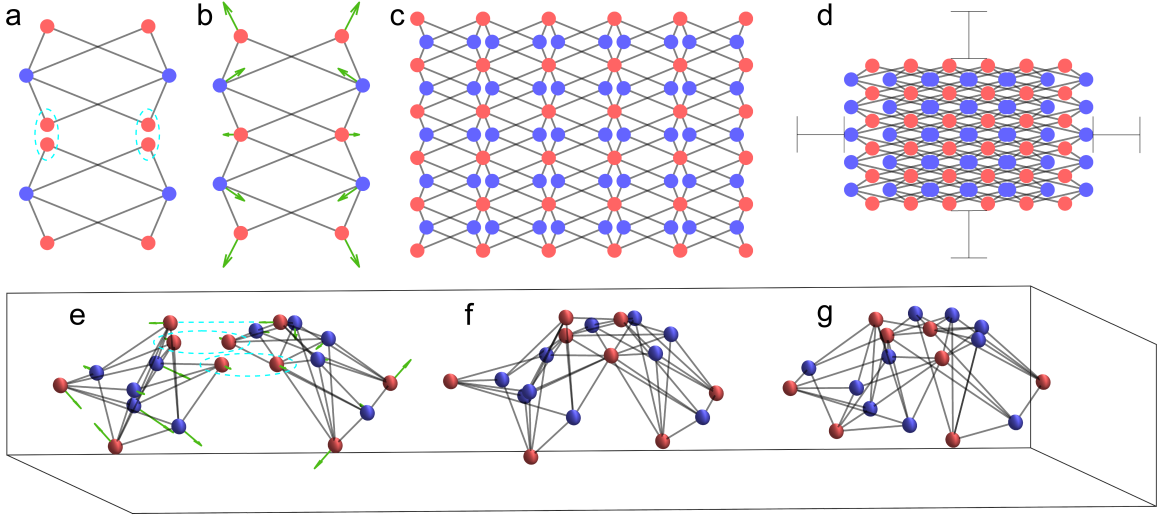


Figure 41: **Combining Network Motions by Merging Nodes & Adding Edges.** (a) Two independent outward moving modules in  $d = 2$  from Fig. 39b, each with a  $D = 1$  conformational motion, where the pairs of nodes circled in cyan are to be merged. (b) The merged network with  $D = 1$ , with the one conformational degree of freedom shown in green arrows. (c) A large network of many coupled modules in the expanded and (d) contracted state, with bars to show the contraction distance from the expanded state. (e) Two independent modules in  $d = 3$ , each with 7 degrees of freedom where the one conformational motion is shown in green arrows, with the nodes to be merged circled in cyan, and the edge to be added shown as a cyan line. (f) The combined network with one conformational motion in the expanded state, and (g) in the contracted state.

Here, we discuss approaches to combine smaller frames for designing metamaterials and allosteric networks. The key is to couple multiple bipartite modules, each with a  $D = 1$  conformational motion, in a way that leaves the entire system with  $D = 1$ .

The natural way to couple networks is to combine nodes. Consider the previously designed

module in  $d = 2$  with  $D = 1$  as an outward motion (Fig. 39b). A system with two of these modules (Fig. 41a) has two independent bodies, each with 3 rigid body motions and a  $D = 1$  conformational motion. By merging two pairs of nodes between modules, we remove two nodes corresponding to four state variables (3 rigid body + 1 conformational), bringing our system to  $D = 1$  while combining our motions (Fig. 41b). This strategy requires that the coupled nodes in each module do not move as a rigid unit. For modules such as these that preserve specific symmetries (see the Appendix in Chapter 7), we can create network lattices with properties such as a negative Poisson ratio (Fig. 41c,d).

We can further remove degrees of freedom by adding extra bonds between module nodes, which becomes necessary in  $d = 3$  as we must remove 7 state variables (6 rigid body and 1 conformational), but coupling two pairs of nodes only removes 6 state variables. For example, consider two modules (Fig. 41e) in  $d = 3$ , each with  $D = 1$ . We remove 6 degrees of freedom by coupling the two overlapping nodes, and the last degree of freedom by adding an extra bond between the modules (Fig. 41f), to yield a coupled network with  $D = 1$  that compounds our motions (Fig. 41g) (see the Appendix in Chapter 7). This long-range coupled conformational regulation of separately synthesized subunits is a hallmark of allostery in enzymes (Changeux and Edelstein, 1998) such as ATCase (Allewell, 1989; Macol et al., 2001b; Cockrell et al., 2013).

## 6.7. Design of Large Displacements & Bistable Networks

Finally, we extend these principles to design networks that achieve large displacements. We first fix some desired initial  $\mathbf{x}_S(0) = \mathbf{x}_S^0$  and final  $\mathbf{x}_S(T) = \mathbf{x}_S^*$  specified node positions as constants, and we treat the initial  $\mathbf{x}_U(0) = \mathbf{x}_U^0$  and final  $\mathbf{x}_U(T) = \mathbf{x}_U^*$  positions of unspecified nodes as variables. Next, we stipulate that the edge lengths at the initial and final positions are equal  $(\mathbf{x}_{S_i}^0 - \mathbf{x}_{U_j}^0)^T (\mathbf{x}_{S_i}^0 - \mathbf{x}_{U_j}^0) = (\mathbf{x}_{S_i}^* - \mathbf{x}_{U_j}^*)^T (\mathbf{x}_{S_i}^* - \mathbf{x}_{U_j}^*)$  with equilibrium length  $l_{ij}^*$ , and rewrite these constraints such that they are linear with respect to variables

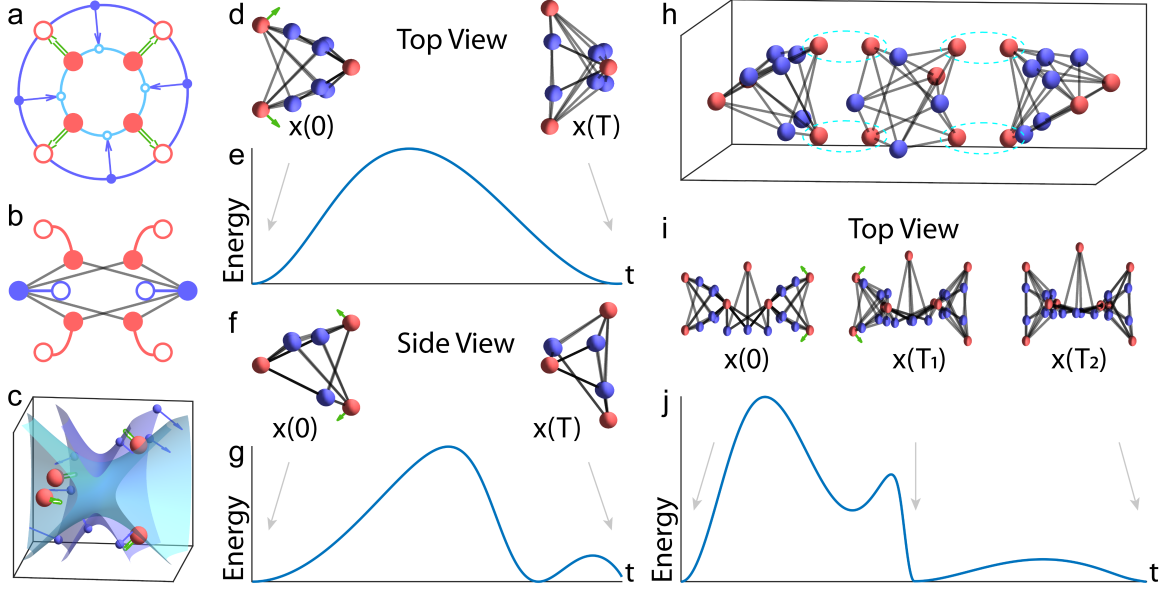


Figure 42: **Designing Finite Motions & Bistable Networks with Cooperativity.** (a) A schematic of the initial ( $\mathcal{M}_0$ , dark blue) and final ( $\mathcal{M}_T$ , light blue) solution spaces in  $d = 2$  for the finite outward motion of four specified nodes with desired initial (solid circle) and final (empty circle) positions. (b) Construction of a network with the  $D = 1$  conformational motion with simulated trajectory. (c) The initial (dark blue) and final (light blue) solution space for four specified nodes with finite motion in hollow green arrows. (d) The top view of the initial and final conformations of a bistable network with  $D = 0$  motions, where the transition is forced from the nodes marked with green arrows by overcoming (e) an energy barrier. (f) A side view of the same network forced from a different set of nodes marked by green arrows by overcoming (g) a different energy barrier. (h) Coupling two of these bistable modules (left, right) through an intermediary network (center) with a  $D = 1$  outward conformational motion by coupling the nodes circled in light blue. (i) An example of cooperativity with transitions from the initial state to intermediary state  $\mathbf{x}(T_1)$ , and then to the final state  $\mathbf{x}(T_2)$  by forcing the nodes in green arrows, where (j) the first transition requires more energy than the second transition.

$\mathbf{x}_{U_j}^0, \mathbf{x}_{U_j}^*$  to yield,

$$\underbrace{\begin{bmatrix} 2\mathbf{x}_{S1}^{0T} & -2\mathbf{x}_{S1}^{*T} & -1 \\ \vdots & \vdots & \vdots \\ 2\mathbf{x}_{Sk}^{0T} & -2\mathbf{x}_{Sk}^{*T} & -1 \end{bmatrix}}_M \underbrace{\begin{bmatrix} \mathbf{x}_{U_j}^0 \\ \mathbf{x}_{U_j}^* \\ c \end{bmatrix}}_v = \underbrace{\begin{bmatrix} \mathbf{x}_{S1}^{0T}\mathbf{x}_{S1}^0 - \mathbf{x}_{S1}^{*T}\mathbf{x}_{S1}^* \\ \vdots \\ \mathbf{x}_{Sk}^{0T}\mathbf{x}_{Sk}^0 - \mathbf{x}_{Sk}^{*T}\mathbf{x}_{Sk}^* \end{bmatrix}}_b, \quad (6.7)$$

with nonlinearity  $c = \mathbf{x}_{Uj}^{0T} \mathbf{x}_{Uj}^0 - \mathbf{x}_{Uj}^{*T} \mathbf{x}_{Uj}^*$ . We then derive the same solution  $\mathbf{v} = W\boldsymbol{\alpha} + \mathbf{v}^*$  as in Eq. 6.4 with quadratic constraint as in Eq. 6.5 with minor modifications (see Methods). Here, we denote  $\mathcal{M}_0$  and  $\mathcal{M}_T$  as all initial and final positions of unspecified nodes satisfying Eq. 6.7. As an example in  $d = 2$ , we illustrate  $\mathcal{M}_0$  in dark blue,  $\mathcal{M}_T$  in light blue, and simulate a network with  $D = 1$  for a large outward motion in four specified nodes (Fig. 42ab).

We can construct bistable networks by first constraining these networks until  $D = 0$ , and then replacing the rigid edges with linear springs such that the network potential energy is

$$E(t) = \frac{1}{2} \sum_{(i,j) \in \mathcal{E}} k_{ij} (l_{ij}(t) - l_{ij}^*)^2.$$

By construction, the initial and final conformations have  $E(0) = E(T) = 0$ , while the  $D = 0$  condition requires an intermediary energy barrier. We illustrate the solution spaces  $\mathcal{M}_0, \mathcal{M}_T$  for four specified nodes (Fig. 42c), and constrain our network by placing nodes along  $\mathcal{M}_0$  until  $D = 0$ . Interestingly, certain networks display asymmetries, such that reaching the final state by forcing one pair of nodes (Fig. 42d) requires overcoming an energy barrier (Fig. 42e) that is different than forcing another pair of nodes (Fig. 42fg).

While these bistable networks already capture more biophysical quantities in protein allostery such as finite conformations and energy barriers, we can further design cooperativity into our networks. Cooperativity is a ubiquitous phenomena in proteins where the binding of a substrate facilitates the subsequent binding of more substrate, and is famously observed in the hemoglobin heterotetramer. We begin with two of our bistable networks, and we couple them to an intermediary network with a  $D = 1$  outward conformational motion by overlapping nodes (Fig. 42h). We first displace the nodes of one module to transition the network conformation from rest  $\mathbf{x}(0)$  to an intermediary bound state  $\mathbf{x}(T_1)$  (Fig. 42i) by overcoming an energy barrier (Fig. 42j). Then, we displace the nodes of the second module to reach the double bound state  $\mathbf{x}(T_2)$ , requiring substantially less energy. Because the network is symmetric, the order of node displacement does not matter.

## 6.8. Discussion

Deciphering principles of control in mechanical systems is of fundamental importance to understanding and optimizing the function of allosteric and cooperative enzymes, auxetics, multistable networks, and tunable metamaterials. Taken together, our work provides fundamental analytic and geometric principles for the construction, characterization, and control of motions and large finite displacements in 2-D and 3-D mechanical frames.

Important prior work in material design has focused on the use of algorithms to tune the infinitesimal responses of mechanical networks and packings (Goodrich et al., 2015; Rocks et al., 2017a). In contrast, our approach provides an analytic characterization of all networks that achieve desired responses in 2 and 3 dimensions for both small and large displacements. Hence, our characterization is complete and invariant to any algorithm, cost function, or initial condition of network topology and geometry. We fully extend these benefits to networks with critical geometries, and design networks with fundamentally finite and sequential properties such as cooperativity.

Important prior theoretical contributions provide valuable conditions for motions of bipartite frames (Whiteley, 1984), or consider the properties and modification of predetermined structures, such as bistabilities of the Miura-ori tessellation (Silverberg et al., 2014a), and topological soft modes at dislocations in Kagome and square lattices (Paulose et al., 2015). Prior work has also sought to systematically enumerate lattices that yield auxetic behavior (Körner and Liebold-Ribeiro, 2015). Importantly, these works require a pre-existing structure, and do not address arbitrary and heterogeneous desired motions. Our approach characterizes the full space of network solutions to achieve these arbitrary motions, and avoids complications associated with local minima and initial network configurations.

Having gained the tools to generate desired motions in complex networks, we begin thinking about applications in physics, biology, medicine, and engineering. One such application may be a bottom-up approach to designing cooperativity and allostery in proteins (Lukin

and Ho, 2004) using known protein structural motifs and simulations to design macroscopic conformational changes. Another application is the design of materials with two independently controllable modes of deformation that behave auxetically or non-auxetically under different perturbations (Lee et al., 2012). These tools could also characterize solution spaces of networks with a desired motion, and use these spaces to search for existing materials with these motions (Dagdelen et al., 2017). Finally, we can design networks to generate precise and complex distributions of spatial forces using few actuators for complex tasks such as grasping in 3 dimensions (Yu Zheng and Wen-Han Qian, 2005). In any application, we can develop a battery of modules that can be coupled to yield even more complex responses, simplifying the network design process into a module-coupling problem.

Importantly, when designing networks with large finite displacements, the choice of unspecified node positions may not allow for a zero energy transition from specified initial to final states. While both states are guaranteed to themselves require 0 energy, and placing unspecified nodes to have  $D = 1$  conformational motions guarantees a finite 0 energy displacement from these states, these 0 energy trajectories may not connect to each other. This phenomena arises as a bifurcation in the kernel of the energy function, and is a fascinating direction of future research.



### 7.1. Defining a Projection from Solution Coordinates to Spatial Coordinates

Our solution space for unspecified node positions and motions is given by linear combinations  $\mathbf{v} = W\boldsymbol{\alpha} + \mathbf{v}^*$  constrained by  $\begin{bmatrix} \boldsymbol{\alpha}^T & 1 \end{bmatrix} Q \begin{bmatrix} \boldsymbol{\alpha} \\ 1 \end{bmatrix} = 0$ . However, when solving for the intersection of solutions for the design of multiple motions  $\dot{\mathbf{x}}_{S1}, \dot{\mathbf{x}}_{S2}$ , we have multiple matrices  $M_1, M_2$ , where the variables  $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2$  are not necessarily represented in the same spatial coordinates. To meaningfully solve for these intersections, we must first transform our solution coordinates into a common space in  $d$  dimensions (e.g.,  $x, y, z$ ). A crucial component of this transformation is the dimension of the coordinate space, given by  $\dim(\mathcal{N}(M)) = m$ .

#### 7.1.1. Case 1: Number of Solution Coordinates: $m = d$

If  $m = d$ , then we have at least  $d$  solution coordinates in  $\boldsymbol{\alpha}$ . To convert to spatially common coordinates, we seek a transformation matrix  $P$  such that  $\begin{bmatrix} \boldsymbol{\beta} \\ 1 \end{bmatrix} = P \begin{bmatrix} \boldsymbol{\alpha} \\ 1 \end{bmatrix}$ . We desire that the  $d$  entries of  $\boldsymbol{\beta}$  correspond to spatial coordinates (e.g.,  $x, y, z$  for  $d = 3$ ). Recall that

$$\mathbf{v} = \begin{bmatrix} \mathbf{x}_{Uj} \\ \dot{\mathbf{x}}_{Uj} \\ c \end{bmatrix} = \begin{bmatrix} W & \mathbf{v}^* \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ 1 \end{bmatrix},$$

such that linear combinations of the first to  $d$  rows correspond to the spatial coordinates we seek as the first  $d$  entries of  $\boldsymbol{\beta}$ . Specifically, we can write

$$B = \begin{bmatrix} I_{m \times m} & 0_{m \times 1} & 0_{m \times 2d-m} \end{bmatrix},$$

as a matrix that isolates the first to  $m = d$  rows via multiplication to yield

$$\begin{bmatrix} \boldsymbol{\beta} \\ 1 \end{bmatrix} = \begin{bmatrix} BW & B\tilde{\mathbf{v}}^* \\ 0_{1 \times m} & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ 1 \end{bmatrix} = P \begin{bmatrix} \boldsymbol{\alpha} \\ 1 \end{bmatrix}$$

With this transformation, we can create a transformed quadratic form

$$\begin{bmatrix} \boldsymbol{\alpha}^T & 1 \end{bmatrix} Q \begin{bmatrix} \boldsymbol{\alpha} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\beta}^T & 1 \end{bmatrix} P^{-T} Q P^{-1} \begin{bmatrix} \boldsymbol{\beta} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\beta}^T & 1 \end{bmatrix} \tilde{Q} \begin{bmatrix} \boldsymbol{\beta} \\ 1 \end{bmatrix} = 0,$$

where the first  $d$  entries of the solution  $\boldsymbol{\beta}$  will be in spatial coordinates.

### 7.1.2. Case 2: Number of Solution Coordinates: $m = d - 1$

If the number of original coordinates is  $m = d - 1$ , then we have one fewer solution dimensions than spatial dimensions, and a direct linear transformation matrix  $P$  is insufficient. We move forward by treating the coordinate of the particular solution  $\mathbf{v}^*$  as a part of the homogeneous solution in  $\mathcal{N}(M)$  such that

$$\mathbf{v} = \begin{bmatrix} \mathbf{x}_{Uj} \\ \dot{\mathbf{x}}_{Uj} \\ c \end{bmatrix} = \begin{bmatrix} W & \mathbf{v}^* \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ \hat{\alpha} \end{bmatrix},$$

where  $\hat{\alpha}$  should equal 1. Then similar to before, we select the spatial coordinates of our solution using matrix

$$B = \begin{bmatrix} I_{d \times d} & 0_{d \times 1} & 0_{d \times d} \end{bmatrix},$$

to get the transformation

$$\begin{bmatrix} \boldsymbol{\beta} \\ \hat{\beta} \end{bmatrix} = B \begin{bmatrix} W & \mathbf{v}^* \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ \hat{\alpha} \end{bmatrix} = P \begin{bmatrix} \boldsymbol{\alpha} \\ \hat{\alpha} \end{bmatrix}.$$

However, because  $\hat{\alpha}$  must be 1, we have the extra constraint that for  $\mathbf{p} \in \mathbb{R}^{d \times 1}$  where  $\mathbf{p} = [0; \dots; 0; 1]$ ,

$$\mathbf{p}^T \begin{bmatrix} \boldsymbol{\alpha} \\ \hat{\alpha} \end{bmatrix} = \hat{\alpha} = \mathbf{p}^T P^{-1} \begin{bmatrix} \boldsymbol{\beta} \\ \hat{\beta} \end{bmatrix} = 1.$$

Hence, our transformation leads to the same form as the previous case

$$\begin{bmatrix} \boldsymbol{\alpha}^T & 1 \end{bmatrix} Q \begin{bmatrix} \boldsymbol{\alpha} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\beta}^T & 1 \end{bmatrix} P^{-T} Q P^{-1} \begin{bmatrix} \boldsymbol{\beta} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\beta}^T & 1 \end{bmatrix} \tilde{Q} \begin{bmatrix} \boldsymbol{\beta} \\ 1 \end{bmatrix} = 0,$$

with the added condition that

$$\mathbf{p}^T P^{-1} \begin{bmatrix} \boldsymbol{\beta} \\ \hat{\beta} \end{bmatrix} = 1.$$

Intuitively, what we have done is artificially extend our solution space to  $d$  coordinates such that our quadratic constraint defines a  $d - 1$  dimensional manifold, and we realize that the true solution space lies at the intersection of this manifold and the  $d - 1$  dimensional hyperplane defined by  $\mathbf{p}^T P^{-1} \begin{bmatrix} \boldsymbol{\beta} \\ \hat{\beta} \end{bmatrix} = 1$ . This way, we can change the coordinates of our original quadratic forms to  $d$  spatial coordinates, and we find the intersection of these quadratics and mathematically well defined hyperplanes.

## 7.2. Characterizing the Set of All Solution Spaces

In the text, we explore several specific examples of placing our unspecified nodes at the intersection of two solution spaces  $\mathbf{x}_{Uj} \in \mathcal{M}_1 \cap \mathcal{M}_2$ , and the resulting multi-mode motions and states of self-stress. The two solution spaces  $\mathcal{M}_1, \mathcal{M}_2$  arose as a result of two independent desired conformational motions  $\dot{\mathbf{x}}_{S1}, \dot{\mathbf{x}}_{S2}$  from particular specified node positions  $\mathbf{x}_S$ . Here, we will characterize the set of solution spaces in a systematic way. Specifically, given our choice of specified node positions  $\mathbf{x}_S$ , we solve for all possible solution spaces  $\mathcal{M}_i$  that

could be generated by any  $\dot{\mathbf{x}}_{S_i}$ .

We begin by placing the solution space coordinates  $\boldsymbol{\alpha}$  into common spatial coordinates  $\boldsymbol{\beta} = \mathbf{x}$  as demonstrated in the previous section. As our intent is to study the intersection of these spaces that only happen meaningfully and non-trivially in  $\dim(\mathcal{M}) = d-1$  dimensions as shown in the main text, we will restrict our discussion to these cases. Then

$$\begin{bmatrix} \boldsymbol{\alpha}^T & 1 \end{bmatrix} Q \begin{bmatrix} \boldsymbol{\alpha} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\beta}^T & 1 \end{bmatrix} \tilde{Q} \begin{bmatrix} \boldsymbol{\beta} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{x}^T & 1 \end{bmatrix} \tilde{Q} \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} = 0,$$

where  $\mathbf{x}$  correspond to the standard coordinate basis elements in  $\mathbb{R}^d$ .

Next, we note a useful property of these quadratic systems. If  $\tilde{Q}_1$  and  $\tilde{Q}_2$  are matrices that satisfy the quadratic constraint for some point  $\mathbf{x}$  such that

$$\begin{bmatrix} \mathbf{x}^T & 1 \end{bmatrix} \tilde{Q}_1 \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{x}^T & 1 \end{bmatrix} \tilde{Q}_2 \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} = 0,$$

then any linear combination  $\tilde{Q} = a_1\tilde{Q}_1 + a_2\tilde{Q}_2$  also satisfies the constraint at those points

$$\begin{aligned} \begin{bmatrix} \mathbf{x}^T & 1 \end{bmatrix} \tilde{Q} \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} &= \begin{bmatrix} \mathbf{x}^T & 1 \end{bmatrix} (a_1\tilde{Q}_1 + a_2\tilde{Q}_2) \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} \\ &= a_1 \begin{bmatrix} \mathbf{x}^T & 1 \end{bmatrix} \tilde{Q}_1 \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} + a_2 \begin{bmatrix} \mathbf{x}^T & 1 \end{bmatrix} \tilde{Q}_2 \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} \\ &= 0. \end{aligned}$$

Finally, we consider all conformational motions in our system of equations

$$\underbrace{\begin{bmatrix} \dot{\mathbf{x}}_{S1}^T & \mathbf{x}_{S1}^T & -1 \\ \vdots & \vdots & \vdots \\ \dot{\mathbf{x}}_{Sn}^T & \mathbf{x}_{Sn}^T & -1 \end{bmatrix}}_M \underbrace{\begin{bmatrix} \mathbf{x}_{Uj} \\ \dot{\mathbf{x}}_{Uj} \\ c \end{bmatrix}}_v = \underbrace{\begin{bmatrix} \mathbf{x}_{S1}^T \dot{\mathbf{x}}_{S1} \\ \vdots \\ \mathbf{x}_{Sn}^T \dot{\mathbf{x}}_{Sn} \end{bmatrix}}_b.$$

Here, we are interested in conformational motions, and thus we remove rigid body translations and rotations that generate trivial solution spaces. Importantly, we note that if  $M$  has full row rank, then there always exists a particular solution to the system of equations. However, with redundancies,  $M$  loses rank, and the motions  $\dot{\mathbf{x}}_S$  must be selected so as to retain a particular solution such that  $\mathbf{b} \in \mathcal{C}(M)$ . Then, each of these  $p$  independent motions satisfying the above equation generates a quadratic matrix  $\tilde{Q}_i$ , and a corresponding solution space  $\mathcal{M}_i$ . Hence, the set of all possible solution spaces given by positions  $\mathbf{x}_S$  is

$$\begin{bmatrix} \mathbf{x}^T & 1 \end{bmatrix} \tilde{Q} \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{x}^T & 1 \end{bmatrix} \left( \sum_{i=1}^p a_i \tilde{Q}_i \right) \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} = 0.$$

Let  $\mathcal{M}_1$  be the solution space corresponding to the desired specified motion  $\dot{\mathbf{x}}_S$  and quadratic matrix  $\tilde{Q}_1$ . Then for any nontrivial linear combination of matrices  $\tilde{Q}^* = \sum_{i=2}^p a_i \tilde{Q}_i$  generating solution space  $\mathcal{M}^*$ , placing unspecified nodes at the intersection  $\mathbf{x}_{Uj} \in \mathcal{M}_1 \cap \mathcal{M}^*$  will serve to generate multiple motions or states of self stress.

As an example, we consider the  $d = 3$  example in the text, with outward motion  $\dot{\mathbf{x}}_S$  and another motion  $\dot{\mathbf{x}}_S^*$ , that generate quadratic matrices  $\tilde{Q}, \tilde{Q}^*$  (Fig. 43a). We can generate another quadratic matrix by taking linear combinations of the first two, that yields different curve geometries (Fig. 43b–c). Importantly, we see that the intersection between the original curve generated by  $\tilde{Q}$  and any linear combination with  $\tilde{Q}^*$  remains constant. We demonstrate this consistency with another independent motion  $\dot{\mathbf{x}}_S^{**}$ .

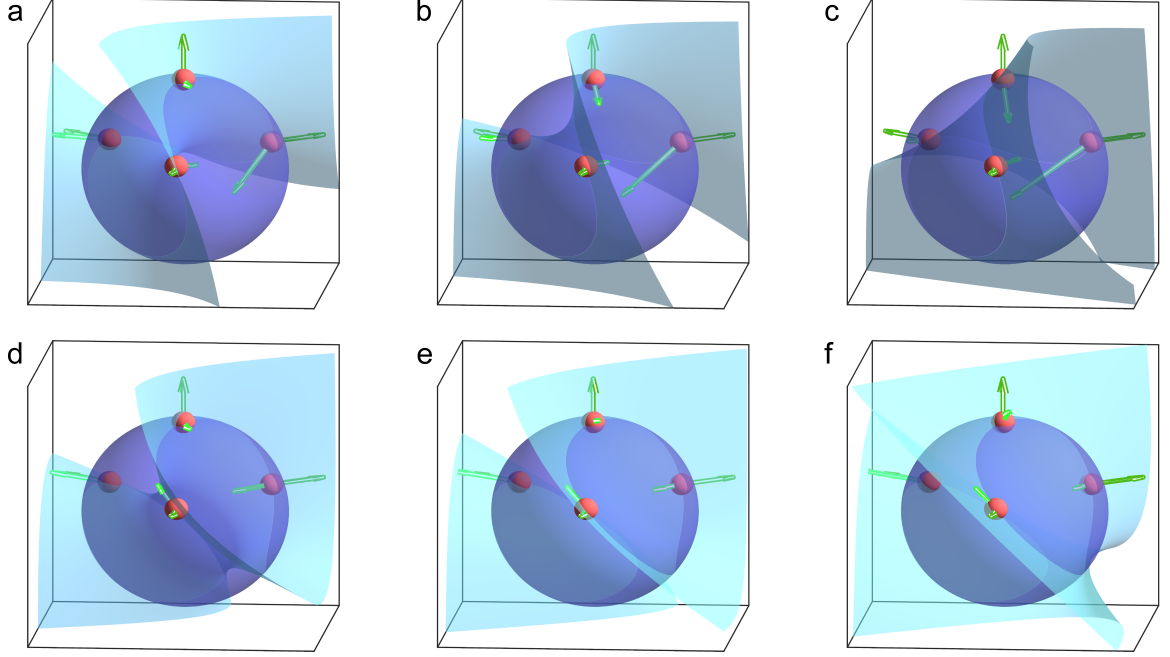


Figure 43: **Solution Space Intersections.** (a) Solution spaces of an outward motion (dark green arrows and blue curve), and another motion (light green arrows and blue curve) for four specified nodes. (b–c) Solution space of a modified outward motion (light green arrows and blue curve) by taking a linear combination of the two motions from panel (a). Note how the intersection of the two curves remains constant. (d–f) A second example of this same process for another independent specified motion.

### 7.3. Expanding on the Judicious Constraint Process

Here we elaborate on the implications of the judicious constraint process components, and we expand on how they might be achieved in a more systematic way.

First, we stipulate that we have  $n > d$  specified nodes that do not all lie on a line in  $d = 2$  or on a plane in  $d = 3$ . The  $n > d$  condition is required because  $n = d = 2$  nodes always lie on a line, and  $n = d = 3$  nodes always lie on a plane. Also, if  $n = d$ , then each unspecified node adds  $d$  state variables and  $d$  constraints, thereby making it impossible to reduce the number of degrees of freedom. The requirement that the nodes do not lie on a line or plane ensures that our nullspace vectors in  $\mathcal{N}(M)$  contribute independently to determining the unspecified node positions (see the section on Dimensionality & Redundancies).

Next, we place unspecified nodes along our solution space  $\mathbf{x}_{U_j} \in \mathcal{M}$  while avoiding states of self-stress. If the solution space dimension is 1 less than the embedding dimension  $\dim(\mathcal{M}) = d - 1$ , then this avoidance can be done in a principled way. First, we find the solution space  $\mathcal{M}_1$  corresponding to the desired specified motion  $\dot{\mathbf{x}}_S$ . Next, we generate the remaining solution spaces  $\mathcal{M}_2, \dots, \mathcal{M}_p$  by generating the remaining  $p - 1$  independent conformational motions by any means (e.g., by randomly generating motions). Finally, because the intersection of these  $\dim(\mathcal{M}) = d - 1$  manifolds has dimension  $\dim(\mathcal{M}_1 \cap \mathcal{M}_i) = d - 2$ , we place each unspecified node  $i$  so as to avoid intersection with a different undesired solution space such that  $\mathbf{x}_{U_i} \notin \mathcal{M}_1 \cap \mathcal{M}_{i+1}$ . Because the solution space intersection  $\mathcal{M}_1 \cap \mathcal{M}_i$  generated by quadratic matrices  $\tilde{Q}_1, \tilde{Q}_i$  is the solution to all nontrivial linear combinations of the quadratic matrices, avoiding overlap with each remaining solution space  $\mathcal{M}_2, \dots, \mathcal{M}_p$  ensures avoiding overlap with any combination of them.

If the solution space dimension is 2 less than the embedding dimension (only feasible for  $\dim(\mathcal{M}) = 1$  in  $d = 3$  dimensions), the situation is more nuanced. This solution space will be a conic section that is the intersection of a quadric surface and a plane (see the section on Defining a Projection to Spatial Coordinates). If 3 unspecified nodes are placed along a plane, then any additional node on that plane can be written as a linear combination of the original 3, yielding redundant constraints and self-stress. To avoid this situation, we can place up to 3 unspecified nodes along  $\mathcal{M}$ , and we can constrain the remaining system by considering subsets of specified nodes that yield solution spaces of dimension 2 (see the section on Avoiding Self-Stress in 3 Dimensions).

Finally, we require that there are no rigid subgraphs, for which there are many efficient algorithms to use after the addition of each unspecified node.

#### 7.4. Rigidity Matrix Dimensions, Spaces, & Degrees of Freedom

The dimensions of the rigidity matrix are given by  $R \in \mathbb{R}^{E \times dN}$ , with  $E$  rows (number of edges) by  $dN$  columns (number of state variables). If our edge constraints are independent,

which is equivalent to saying that the rows of  $R$  are linearly independent, then the total number of degrees of freedom in our system is given by  $N_{DOF} = dN - E$ . Recall that any rigid body contains  $d(d+1)/2$  rigid body motions as degrees of freedom. Then the number of degrees of freedom must be at least  $N_{DOF} \geq d(d+1)/2$ . Importantly, for the edges to be independent, we must have at least  $d(d+1)/2$  more state variables than edges, such that

$$dN \geq E + d(d+1)/2,$$

causing  $R$  to have more columns than rows, and causing the rank of  $R$  to become  $\text{rank}(R) = E$ .

As a result, if the edges are independent such that the rows of  $R$  are linearly independent, then  $\text{rank}(R) = E$ , and the dimension of the nullspace  $\mathcal{N}(R)$  is given by the difference between the number of state variables and constraints, or  $\dim(\mathcal{N}(R)) = dN - E$ . If the rows of  $R$  are not linearly independent, then the rigidity matrix loses rank  $\text{rank}(R) < E$ , and the nullspace increases dimension such that  $\dim(\mathcal{N}(R)) > dN - E$ . We note that the nullspace dimension can never be *less* than  $dN - E$  as long as  $dN - E > 0$ .

## 7.5. Implications of States of Self-Stress in Infinitesimal & Finite Motions

Here, we explore in more detail states of self-stress by studying the rigidity matrix in higher-order rigidity conditions, and the set of all solution spaces. Intuitively, self-stress occurs when a set of edge constraints are *redundant*, such that the constraint conditions overlap. A trivial example is to place two edges between the same pair of nodes corresponding to two identical rows in  $R$ . The more complex scenario we face is when multiple edges between different node pairs conspire to make one row of  $R$  a linear combination of the other rows. In either case, the rigidity matrix loses rank, and the nullspace dimension is larger than expected by constraint counting  $\dim(\mathcal{N}(R)) > dN - E$ .

Self-stress is most commonly encountered in overdetermined systems with more edges than



needed for rigidity such that the system bears internal forces. The unique scenario we face is self-stress in underdetermined systems with fewer edges than needed for rigidity. To explore these special cases, we recall that given specified node positions  $\mathbf{x}_S$ , the set of all solution spaces  $\mathcal{M}_1, \dots, \mathcal{M}_p$  are given by linear combinations of quadratic matrices  $\tilde{Q} = a_1\tilde{Q}_1 + \dots + a_p\tilde{Q}_p$  from all independent specified conformational motions satisfying the constraint equations. Here, we reiterate the implication of self-stress in the infinitesimal regime, followed by an exploration of motions that extend to the finite regime.

#### 7.5.1. Self-Stress in the Infinitesimal Regime

Suppose we begin with a set of specified nodes with positions  $\mathbf{x}_S$  and desired motion  $\dot{\mathbf{x}}_S$ , and we place unspecified nodes  $\mathbf{x}_U$  and corresponding edges along solution space  $\mathcal{M}$  satisfying linear constraints Eq. 6.1 such that  $dN - E = d(d+1)/2 + 1$ . As a result, we know that the desired motion  $\dot{\mathbf{x}}_S$  is contained in a nullspace vector  $\dot{\mathbf{x}} \in \mathcal{N}(R)$  by design, and we expect  $\dim(\mathcal{N}(R)) = d(d+1)/2 + 1$ . The method fails when there is a set of redundant edges such that  $\text{rank}(R) < E$ , and the nullspace dimension is greater than expected when  $\dim(\mathcal{N}(R)) > dN - E$ , such that there is at least one other independent vector in the nullspace  $\dot{\mathbf{x}}^* \in \mathcal{N}(R)$ . This vector  $\dot{\mathbf{x}}^*$  corresponds to a second unintentional infinitesimal specified node motion  $\dot{\mathbf{x}}_S^*$  that generates its own solution space  $\mathcal{M}^*$ , and any linear combination of  $\dot{\mathbf{x}}, \dot{\mathbf{x}}^*$  are in the nullspace  $b_1\dot{\mathbf{x}} + b_2\dot{\mathbf{x}}^* \in \mathcal{N}(R)$ .

#### 7.5.2. Self-Stress in the Finite Regime

Recall that the rigidity matrix  $R = R(\mathbf{x})$  is the evaluation of the first-order time derivative of the nonlinear edge constraints  $\dot{\mathbf{g}}(\mathbf{x}, \dot{\mathbf{x}}) = \mathbf{0}$  at a particular node position  $\mathbf{x}$  (see the section on Finite *versus* Infinitesimal Motions). If our node and edges generate self-stress at  $\mathbf{x}$ , then the rigidity matrix loses rank, such that we have more conformational motions in the nullspace,  $\dot{\mathbf{x}}, \dot{\mathbf{x}}^* \in \mathcal{N}(R)$  than expected. Further, because we can write any  $n$ th

derivative of the positional constraints as

$$R(\mathbf{x})\mathbf{x}^{(n)} = \mathbf{f}(\mathbf{x}, \dots, \mathbf{x}^{(n-1)}),$$

if the rigidity matrix does not have full rank, then there may not exist a solution of  $\mathbf{x}^{(n)}$  to satisfy the derivative condition. For a particular set of  $\mathbf{x}, \dots, \mathbf{x}^{(n-1)}$ , if this scenario ever occurs, then that set of positions, motions, and further derivatives are not finite.

Here we outline and demonstrate the utility of a second order derivative test. Specifically, given positions  $\mathbf{x}$  and motions  $\dot{\mathbf{x}}$  that satisfy the first-order constraint  $\dot{\mathbf{g}}(\mathbf{x}, \dot{\mathbf{x}}) = R(\mathbf{x})\dot{\mathbf{x}} = \mathbf{0}$ , we must find a second derivative  $\ddot{\mathbf{x}}$  such that

$$R(\mathbf{x})\ddot{\mathbf{x}} = -R(\dot{\mathbf{x}})\dot{\mathbf{x}}.$$

We can test if the right-hand side is in the columnspace of the rigidity matrix  $R(\dot{\mathbf{x}})\dot{\mathbf{x}} \in \mathcal{C}(R(\mathbf{x}))$  by first computing vectors in the left nullspace of the rigidity matrix  $\mathbf{w} \in \mathcal{N}(R^T(\mathbf{x}))$ , and projecting them onto the right-hand side. If the right-hand side exists in the columnspace of  $R(\mathbf{x})$ , then  $\mathbf{w}^T R(\dot{\mathbf{x}})\dot{\mathbf{x}} = 0$  for all  $\mathbf{w}$ .

Our situation of self-stress adds some complexity to this requirement, because we must not only test any one motion in the nullspace  $\dot{\mathbf{x}} \in \mathcal{N}(R)$ , but all linear combinations of motions because in our states of self-stress, there are guaranteed to be at least the desired specified motion  $\dot{\mathbf{x}}_S$ , and the unintentional motion  $\dot{\mathbf{x}}_S^*$ . In the general case, let  $\dot{X} = [\dot{\mathbf{x}}^1, \dots, \dot{\mathbf{x}}^k]$  be a matrix whose columns are the linearly independent conformational motions in the nullspace  $\mathcal{N}(R)$ , and let  $\dot{\mathbf{x}} = \dot{X}\boldsymbol{\gamma}$  be a linear combination of these motions. Then for each vector in the left nullspace  $\mathbf{w} \in \mathcal{N}(R^T)$ , we must satisfy

$$\mathbf{w}^T R(\dot{\mathbf{x}})\dot{\mathbf{x}} = \mathbf{w}^T R(\gamma_1 \dot{\mathbf{x}}^1 + \dots + \gamma_k \dot{\mathbf{x}}^k)(\gamma_1 \dot{\mathbf{x}}^1 + \dots + \gamma_k \dot{\mathbf{x}}^k) = 0.$$

Because  $R(\dot{\mathbf{x}})$  only contains first-order elements of its arguments, it is linear in its arguments,

thereby giving us the property  $R(\gamma_1 \dot{\mathbf{x}}^1 + \gamma_2 \dot{\mathbf{x}}^2) = \gamma_1 R(\dot{\mathbf{x}}^1) + \gamma_2 R(\dot{\mathbf{x}}^2)$ . Then we write

$$\begin{aligned}
\mathbf{w}^T R(\dot{\mathbf{x}}) \dot{\mathbf{x}} &= \mathbf{w}^T \left( \gamma_1 R(\dot{\mathbf{x}}^1) + \cdots + \gamma_k R(\dot{\mathbf{x}}^k) \right) \dot{X} \gamma \\
&= \gamma_1 \mathbf{w}^T R(\dot{\mathbf{x}}^1) \dot{X} \gamma + \cdots + \gamma_k \mathbf{w}^T R(\dot{\mathbf{x}}^k) \dot{X} \gamma \\
&= \begin{bmatrix} \gamma_1 & \cdots & \gamma_k \end{bmatrix} \begin{bmatrix} \mathbf{w}^T R(\dot{\mathbf{x}}^1) \dot{X} \gamma \\ \vdots \\ \mathbf{w}^T R(\dot{\mathbf{x}}^k) \dot{X} \gamma \end{bmatrix} \\
&= \gamma^T \begin{bmatrix} \mathbf{w}^T R(\dot{\mathbf{x}}^1) \dot{X} \\ \vdots \\ \mathbf{w}^T R(\dot{\mathbf{x}}^k) \dot{X} \end{bmatrix} \gamma \\
&= \gamma^T \underbrace{\begin{bmatrix} \mathbf{w}^T R(\dot{\mathbf{x}}^1) \dot{\mathbf{x}}^1 & \cdots & \mathbf{w}^T R(\dot{\mathbf{x}}^1) \dot{\mathbf{x}}^k \\ \vdots & \ddots & \vdots \\ \mathbf{w}^T R(\dot{\mathbf{x}}^k) \dot{\mathbf{x}}^1 & \cdots & \mathbf{w}^T R(\dot{\mathbf{x}}^k) \dot{\mathbf{x}}^k \end{bmatrix}}_S \gamma \\
&= 0.
\end{aligned}$$

Here, we require some linear combination  $\gamma$  such that  $\gamma^T S \gamma = 0$  for all  $\mathbf{w} \in \mathcal{N}(R(\mathbf{x}))$ . We also note that due to the property  $R(\mathbf{v})\mathbf{u} = R(\mathbf{u})\mathbf{v}$ , matrix  $S$  is symmetric and therefore has real eigenvalues. Hence, if  $S$  is positive (or negative) definite such that it only contains positive (or negative) eigenvalues, then a solution cannot exist, and the system has no finite deformable motions. This type of network is called *pre-stress stable*. Examples of the design of these networks in  $d = 2$  and  $d = 3$  are shown in the main text.

The alternative is that a solution to this constraint exists, and that the solution takes the form of some  $k - 1$  dimensional quadratic solution. Because  $S$  is symmetric, we can always find an orthonormal diagonalization  $S = PDP^T$  with  $P$  as a square matrix with column eigenvectors, and  $D$  a diagonal matrix with corresponding eigenvalues. Then with a variable

basis change  $\delta = P^T \gamma$ , we rewrite the condition as

$$\mathbf{w}^T R(\dot{\mathbf{x}}) \dot{\mathbf{x}} = \delta^T D \delta = \sum_{i=1}^k \lambda_i \delta_i^2 = 0,$$

and because  $\gamma = P\delta$ , we substitute to get  $\dot{\mathbf{x}} = \dot{X}\gamma = \dot{X}P\delta$  as valid motions to second-order.

## 7.6. Designing & Analyzing Finite Motions in Networks with Self-Stress

Here we apply the ideas of second-order rigidity testing to some examples. We begin with the examples in the main text to demonstrate pre-stress stability, followed by an example of a network that sits at the branching point of two finitely deformable motions.

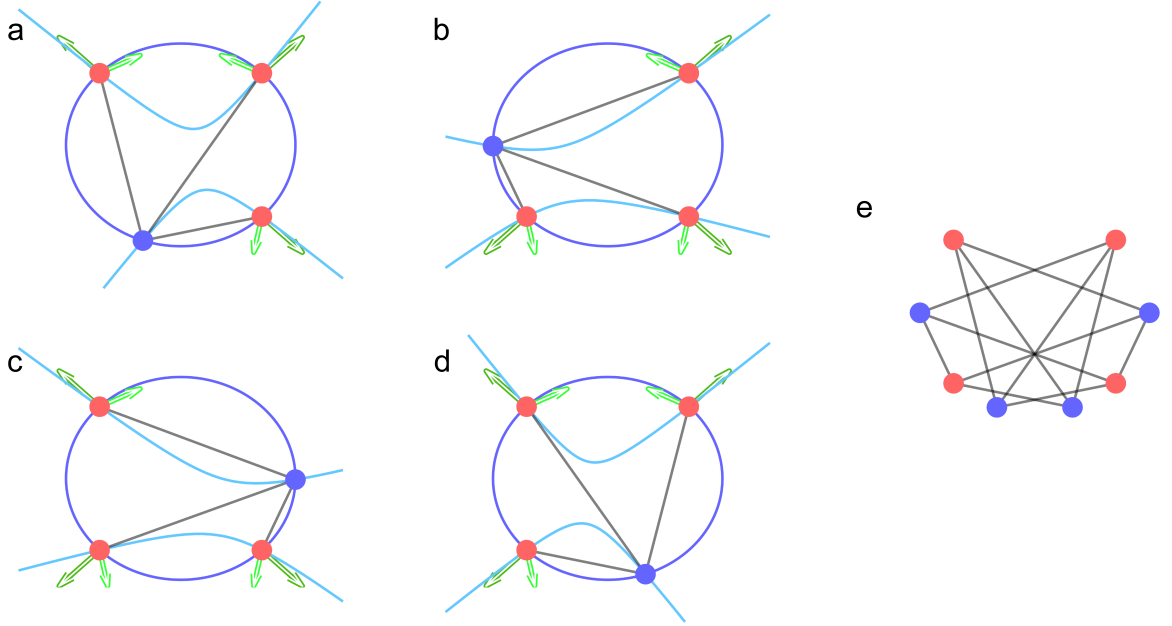


Figure 44: **Constructing a Network that is Pre-Stress Stable.** (a–d) A schematic of two stipulated motions (light and dark green arrows) corresponding to two solution spaces (light and dark blue curves) for all subsets of 3 specified nodes (red) for a 4 specified node network, with a connected unspecified node (blue) at their intersection. (e) The full network with  $D = 1$  expected conformational motions, but 0 finitely deformable ones due to pre-stress stability.

In this example, we have four specified nodes with a radially outward motion shown in dark-green arrows, and a second motion shown in bright-green arrows. We then select all

sets of 3 specified nodes, compute the solution spaces for each of these sets, and place an unspecified node connected to these 3 specified nodes at their intersection (Fig. 44a–d). This construction yields the final network (Fig. 44e) with 16 state variables and 12 edges for an expected conformational motion. However, because the unspecified nodes sit at the intersection of solution spaces of two motions, we have self-stress. The eigenvalues of matrix  $S$  for this system are all positive, and thereby  $\gamma^T S \gamma \neq 0$ .

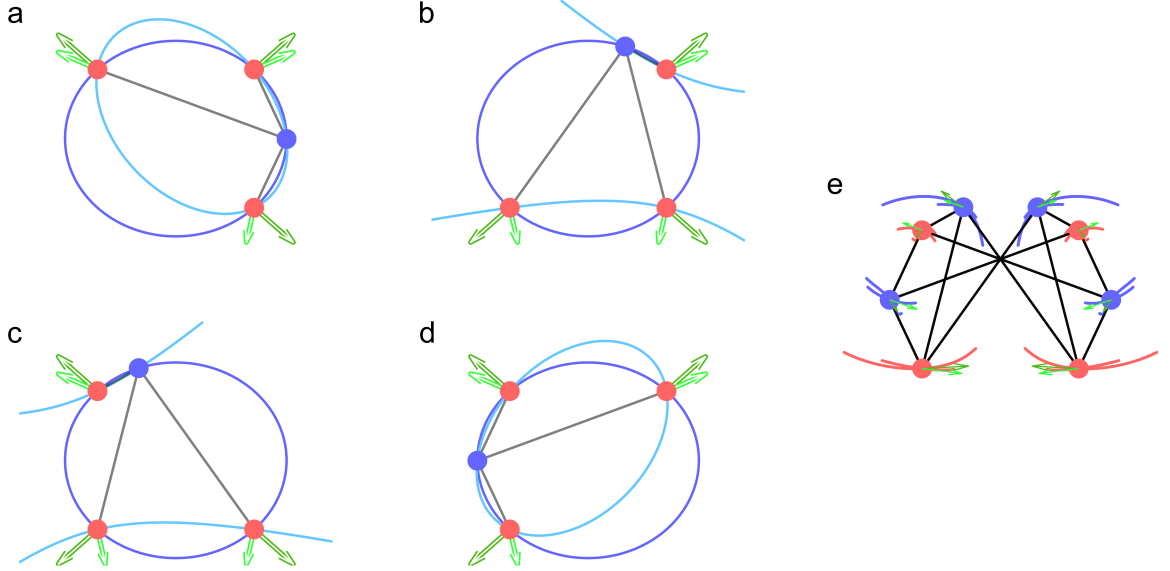


Figure 45: **Constructing a Network at the Intersection of Branching.** (a–d) A schematic of two stipulated motions (light and dark green arrows) corresponding to two solution spaces (light and dark blue curves) for all subsets of 3 specified nodes (red) for a 4 specified node network, with a connected unspecified node (blue) at their intersection. (e) Full network with two possible finitely deformable trajectories shown in red and blue lines. The motions that serve as solutions to the second-order equations are shown in bright and dark green arrows.

Next, we change the stipulated motion of the light-green arrows, and we place unspecified nodes at the intersection of solution spaces for all subsets of 3 specified nodes (Fig. 45a–d), yielding the final network Fig. 45e. In this case, there are two possible branched trajectories that serve as valid finite motions, but linear combinations of them are *not* allowed. Here, the eigenvalues of  $S$  are positive and negative, and the solutions  $\dot{\mathbf{x}} = \dot{X}\gamma$  to the condition  $\gamma^T S \gamma = 0$  are shown in bright and dark green arrows, while the finite simulated trajectories that preserve edge lengths are shown in red and blue lines.

## 7.7. Combining Networks with Repeating Modules

In the main text, we demonstrate the coupling of modules that preserve symmetries in motions to create larger networks. Here, we will outline how this combining is achieved, and we will discuss conditions for when these combined motions extend to the finite regime.

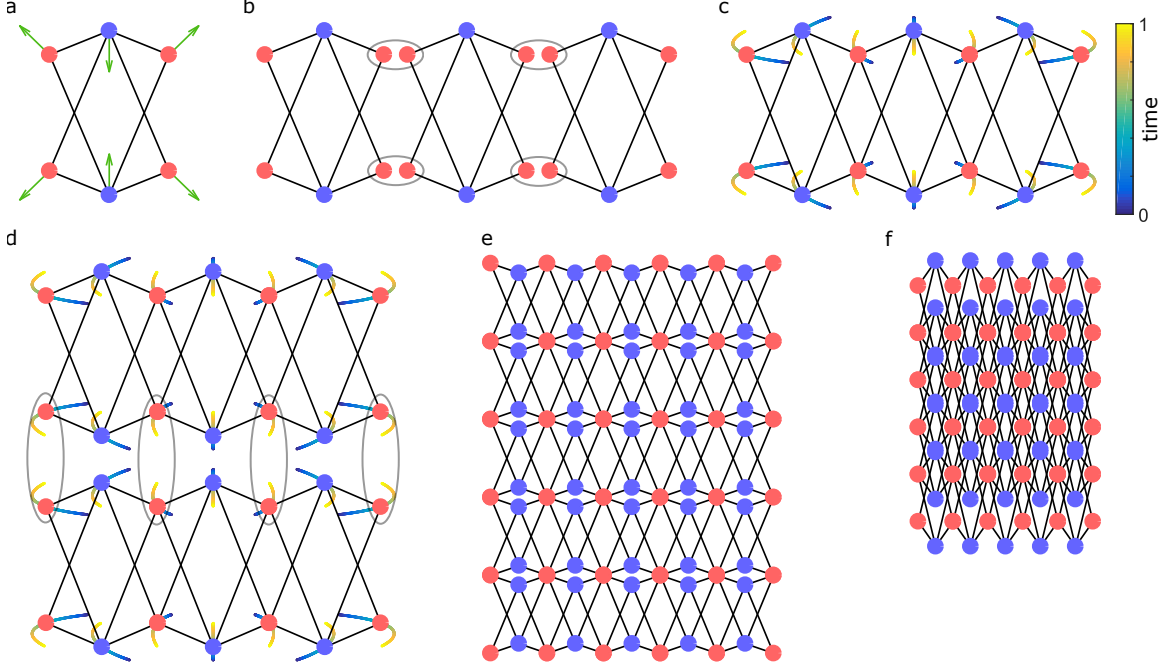


Figure 46: **Combination of Identical Modules with Nonlinear Symmetries Through Node Merging.** (a) A single module in 2 dimensions with 4 specified nodes (red), 2 unspecified nodes (blue), with one non-rigid body degree of freedom (green arrows). (b) Three replicates of the same module placed side-by-side, with the nodes to be merged grouped in the gray curves. (c) Nonlinear motion of the combined network after merging grouped nodes, where the only non-rigid body motion is traced from blue to yellow for each node. (d) Two of these composite networks, aligned side-by-side with nodes to be merged grouped in the gray curves. (e) Full composite network with 4 horizontal and 4 vertical replicate modules with one non-rigid body degree of freedom in the expanded form, and (f) in the contracted forms.

To begin, we consider a simple module (Fig. 46a) in 2 dimensions with 4 specified nodes, 2 unspecified nodes, and 1 non-rigid body degree of freedom shown in green arrows. Notice that this module has two symmetries: one along the horizontal axis, and one along the vertical axis. We can replicate this module along one of these directions (Fig. 46b), and we couple their specified nodes according to the gray curves. Note that this system of 3

modules has  $4 \times 3 = 12$  degrees of freedom, and by grouping the two specified nodes into one node, we remove 4 nodes and  $4 \times 2 = 8$  state variables to yield  $12 - 8 = 4$  degrees of freedom. We show this composite network (Fig. 46c), with the one non-rigid body degree of freedom shown in the full non-linear trajectory with curves for each node parameterized by a time variable from blue to yellow.

We can then replicate this composite network (Fig. 46d), and we notice that for the full non-linear conformational response, the grouped red specified node motions overlap. What we mean here is that as this time parameter is varied from 0 to 1, the x-coordinates of each grouped pair of nodes are equivalent, and the y-coordinates of each grouped pair of nodes is only offset by a single constant  $c(t)$  across all groups, which is simply a rigid body translation. Alternatively, we can say that if we were to combine the node pairs in each group, we only require the addition of rigid body motions to one composite's nonlinear trajectory to exactly follow the other composite's trajectory in the grouped nodes. As an example, we replicate the single module in Fig. 46a four times horizontally, and four times vertically, to create a networked sheet with one non-rigid body degree of freedom that we show in the expanded form (Fig. 46e), and in the contracted form (Fig. 46f). Hence, through the simple replicating and merging of simple modules that preserve certain symmetries, we can create materials that replicate the behavior of one module on a larger scale.

#### *7.7.1. Applying the Higher-Order Derivative Test for Finite Motion*

To complement this analysis, we consider network combination from the perspective of rigidity matrix spaces and higher-order derivative tests to avoid having to perform a point-by-point match of simulated finite trajectories. To begin, we consider the combination of modules in this node-wise fashion, and we note that each module is designed to have only one conformational motion (Fig. 46a). By coupling the modules in node pairs (Fig. 46b), we generate a larger network with one combined conformational motion (Fig. 46c). In general, the preservation of instantaneous conformational motions in node coupling is the first re-

quirement for finite motion. Importantly, because each module only has one conformational motion, if many modules are coupled together such that a motion in one module excites a motion in another, then the combined network has at most one conformational motion.

Next we consider extensions to the finite case. Given a combined network with positions  $\mathbf{x}$  and infinitesimal conformational motion  $\dot{\mathbf{x}} \in \mathcal{N}(R(\mathbf{x}))$ , we test second-order rigidity

$$R(\mathbf{x})\ddot{\mathbf{x}} = -R(\dot{\mathbf{x}})\dot{\mathbf{x}},$$

by finding all vectors in the left nullspace  $V = \mathcal{N}(R(\mathbf{x})^T)$ , and by checking whether there exists a second positional derivative  $\ddot{\mathbf{x}}$  that satisfies the above condition by searching for nontrivial projections onto the left nullspace. If  $V^T R(\dot{\mathbf{x}})\dot{\mathbf{x}} = \mathbf{0}$ , then a solution  $\ddot{\mathbf{x}}$  exists. Otherwise, the motion  $\dot{\mathbf{x}}$  does not extend finitely, and our system is pre-stress stable. Of course, to guarantee motion, we must check all infinite derivatives. However, this condition has the considerable value of identifying networks that are second-order rigid.

Finally, we note that this test is unnecessary when adding one module at a time to one cluster. In the case of  $d = 2$ , the coupling is very simple, where two nodes of one module merge with two nodes of another. Each module has 4 degrees of freedom (3 rigid body and 1 conformational), and merging two nodes removes four of these degrees of freedom such that the total system is left with 3 rigid body, 1 conformational motion. Similarly, in  $d = 3$ , the merging of 2 nodes and adding of one edge removes 7 degrees of freedom (6 rigid body and 1 conformational). This process removes the appropriate number of degrees of freedom without generating self-stress, and can be performed with no reservation, and the resulting network will always have 1 finite conformational motion.

We must be careful when performing couplings between two modules at more than 2 nodes (Fig. 46d), as combining more than 2 nodes removes more than 4 state variables, and we must ensure that the states of self-stress generated by this process are orthogonal to



higher-order motion derivatives. That is, to satisfy

$$R(\mathbf{x})\mathbf{x}^{(n)} = \mathbf{f}(\mathbf{x}, \dots, \mathbf{x}^{(n-1)}),$$

we require that the self stress of the system  $V = \mathcal{N}(R(\mathbf{x})^T)$  be orthogonal such that

$$V^T \mathbf{f}(\mathbf{x}, \dots, \mathbf{x}^{(n-1)}) = \mathbf{0}.$$

### 7.7.2. Combining Networks Through Judicious Constraint Placement

Consider a set of  $|\mathcal{V}_S| = n$  specified nodes embedded in  $d$  dimensions with coordinates  $\mathbf{x}_S \in \mathbb{R}^{dn}$ , and with desired displacements  $\dot{\mathbf{x}}_S \in \mathbb{R}^{dn}$ . In general, the solution space of an unspecified node  $j$  with  $2d$  variables (position  $\mathbf{x}_{Uj}$  and motion  $\dot{\mathbf{x}}_{Uj}$ ) constrained by connections to  $k$  specified nodes has dimension  $2d - k$ . For  $n > 2d$ , we generally cannot place unspecified nodes connected to all  $n$  specified nodes in a manner that preserves desired motions  $\dot{\mathbf{x}}_S$ .

Instead, we can partition the  $n$  nodes into  $p$  non-overlapping *primary* modules  $P_i \subseteq \mathcal{V}_S$  where  $|P_i| \leq 2d$  nodes and  $P_i \cap P_j = \emptyset$ , and we judiciously constrain each module individually through the judicious placement of unspecified nodes to have  $d(d+1)/2 + 1$  degrees of freedom. Then, we can couple these modules by constraining a second set of *coupling* modules  $C_k \subset \{P_i \cup P_j\}$  such that  $|C_k| \leq 2d$ , while ensuring the entire network has the necessary number of degrees of freedom to achieve the desired motion.

As an example in  $d = 2$ , we partition a set of 6 specified nodes with desired motions (Fig. 47a) into two primary modules  $P_1, P_2$  and two coupling modules  $C_1, C_2$  (Fig. 47b) where  $|P_1| = |P_2| = |C_1| = |C_2| = 3$ . We judiciously constrain the primary modules to have 4 degrees of freedom, and we also judiciously constrain coupling modules  $C_1, C_2$  (Fig. 47c) until the final network has 4 degrees of freedom, with our desired motion as the one non-rigid body motion (Fig. 47d). As another example in  $d = 3$ , we partition a set of

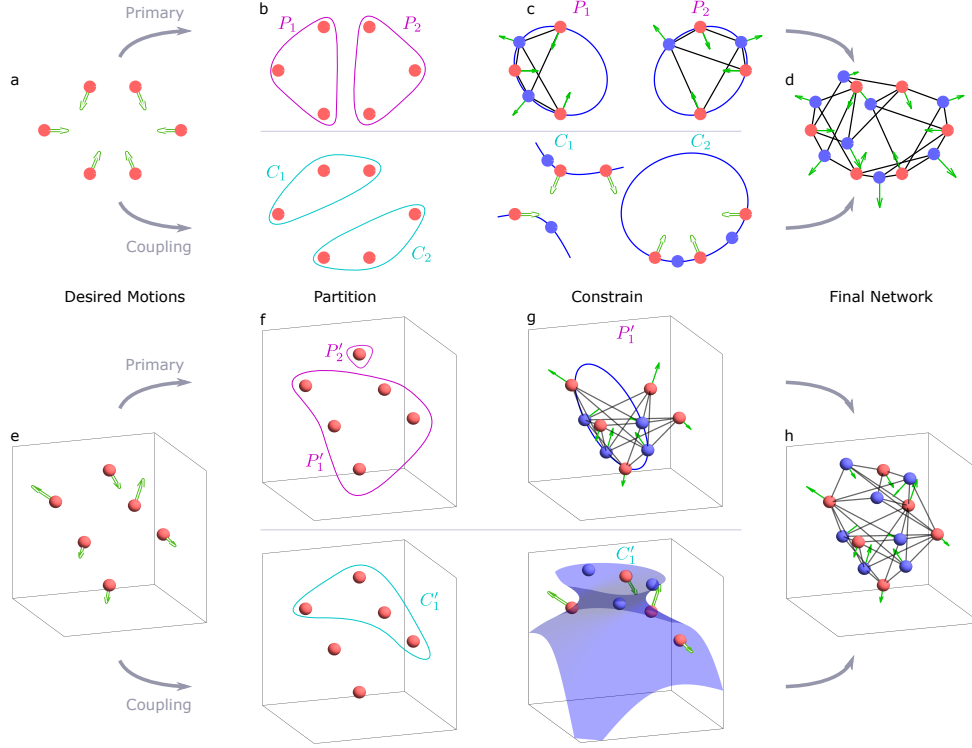


Figure 47: **Construction of Large Network Motions Through the Judicious Coupling Between Non-Intersecting Modules.** (a) Example in  $d = 2$  of six specified node positions (red) and motions (hollow green arrows) with no solution for the placement of an unspecified node. (b) Partitioning of specified nodes into two primary modules ( $P_1, P_2$ , purple curve), coupled by two coupling modules ( $C_1, C_2$ , light blue curve). (c) Construction of primary modules  $P_1, P_2$  through judicious constraint placement such that both primary modules have four degrees of freedom, with the non-rigid motion in solid green arrows, followed by the judicious constraining of the coupled modules  $C_1, C_2$  by placing unspecified nodes (blue circles) on the solution space (blue curve) to bring  $D$  of the total system (d) down to  $D=1$ , with the only non-rigid body motion shown in solid green arrows. (e) Example in  $d = 3$  of six specified nodes with no unspecified solution space, (f) partitioned into two primary modules ( $P'_1, P'_2$ , purple curve) and one coupling module ( $C'_1$ , light blue curve). (g) Judicious constraint construction of primary module  $P'_1$  to seven degrees of freedom by placing four unspecified nodes along the unspecified solution space, and judicious constraint placement of the coupling module  $C'_1$  with a two dimensional solution space (blue surface) to yield (h) the constructed network with the true and only non-rigid body motion (solid green arrows).

6 specified nodes (Fig. 47e) into two primary modules  $P'_1, P'_2$  and one coupling module  $C'_1$  (Fig. 47f), where  $|P'_1| = 5$ ,  $|P'_2| = 1$ , and  $|C'_1| = 4$ . We first judiciously constrain  $P'_1$  along the unspecified solution space until it has  $D = 1$ ; then we constrain the coupling module  $C'_1$

(Fig. 47g) until the final network (Fig. 47h) has  $D = 1$ , with the desired motion as the only non-rigid body degree of freedom. We see that by judiciously constraining these primary and coupling modules, we can design arbitrary motions in large networks. If the modules preserve some symmetries in their motions, this coupling can be performed much more efficiently through the combining of nodes to create materials that replicate the module motion on a larger scale. We note that this procedure is simply extended to the design of networks with multiple motions  $\dot{\mathbf{x}}_{S1}, \dot{\mathbf{x}}_{S2}$  by constraining the primary modules and the full network to have  $d(d+1)/2 + 2$  degrees of freedom.

### 7.8. Avoiding States of Self Stress in 3 Dimensions

One crucial condition to guarantee finitely deformable motions is to avoid states of self stress during the judicious constraint process. A peculiar situation arises when designing networks with 5 specified node positions and motions  $|\mathcal{V}_S| = 5$  in  $d = 3$ . In general, for 5 specified nodes with independent motions, we have  $\dim(\tilde{A}) = 2$ , with a one dimensional solution space that is the intersection of a quadric surface (defined by the first four nodes) and a plane (defined by the last node). Hence, all unspecified nodes in  $\mathcal{V}_U$  must be placed coplanar to each other, which creates a bipartite network that provably has at least two infinitesimal motions. These motions mean that if we judiciously constrain our 5 specified nodes to try to achieve a total  $D = 1$  conformational motion, we end up with 6 rigid body motions, 2 infinitesimal motions, and 1 state of self stress.

For example, consider one of the modules in the main text in  $d = 3$  concerning network combination. The desired positions and motions of the specified nodes are

$$\mathbf{x}_1 = \begin{bmatrix} 2 \\ 0 \\ 1 \end{bmatrix}, \mathbf{x}_2 = \begin{bmatrix} 1 \\ 0 \\ 2 \end{bmatrix}, \mathbf{x}_3 = \begin{bmatrix} 3 \\ 0 \\ 3 \end{bmatrix}, \mathbf{x}_4 = \begin{bmatrix} 3.5 \\ -1 \\ 2.5 \end{bmatrix}, \mathbf{x}_5 = \begin{bmatrix} 3.5 \\ 1 \\ 2.5 \end{bmatrix},$$

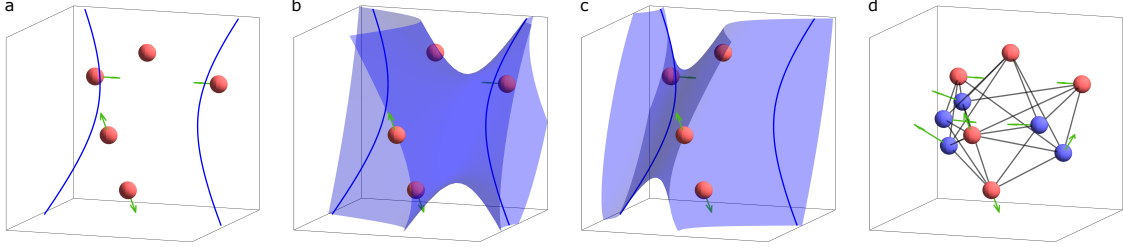


Figure 48: **Non-Planar Judicious Constraint of 5 Nodes in  $d = 3$ .** (a) Planar 1-dimensional solution space (blue curves) for the position and motion of an unspecified node connected to 5 independent specified nodes. (b) 2-dimensional solution quadric surface (blue) for one subset of four of the five specified nodes, and (c) for a different subset. (d) Final constrained network with 7 degrees of freedom and no states of self stress, with the finitely deformable conformational motion in green arrows.

$$\dot{\mathbf{x}}_1 = \begin{bmatrix} -0.8 \\ 0 \\ 0.8 \end{bmatrix}, \dot{\mathbf{x}}_2 = \begin{bmatrix} 0.8 \\ 0 \\ -0.8 \end{bmatrix}, \dot{\mathbf{x}}_3 = \begin{bmatrix} -0.8 \\ 0 \\ 0 \end{bmatrix}, \dot{\mathbf{x}}_4 = \begin{bmatrix} 0 \\ -0.8 \\ 0 \end{bmatrix}, \dot{\mathbf{x}}_5 = \begin{bmatrix} 0 \\ 0.8 \\ 0 \end{bmatrix}.$$

The motions were scaled to 0.8 for purely aesthetic reasons so that the figure arrows would not overlap. Here, we solve for and visually demonstrate that the 1-dimensional solution space lies along a plane (Fig. 48a), such that even if we added  $m = 4$ ,  $E = 20$  to theoretically  $N_D = 3(5 + 4) - 20 = 7$ , the generated state of self stress would not guarantee our desired motion as the sole finitely deformable motion.

In response, we can add three coplanar unspecified nodes connected to all five specified nodes along the 1 dimensional solution space (Fig 48a) to yield  $N_D = 3(5 + 3) - 15 = 9$ . Then we can remove the final two degrees of freedom by judiciously constraining two separate subsets of 4 specified nodes (Fig. 48bc) along the quadric surface of solutions that are not coplanar to the initial 3 unspecified nodes, to get our final network  $N_D = 7$ , with the one desired finitely deformable conformation degree of freedom (Fig. 48d).

## 7.9. Consideration of Non-Bipartite Edges in Modules

Here we discuss the placement of non-bipartite edges in our module design. First, we consider edge  $k$  between specified nodes  $(i, j)$ , that must satisfy

$$\dot{g}_k = (\mathbf{x}_{Si} - \mathbf{x}_{Sj})^T (\dot{\mathbf{x}}_{Si} - \dot{\mathbf{x}}_{Sj}) = 0,$$

and we realize that an edge can only exist if the net motion is perpendicular to the bond between them. Because the specified node positions and motions are set as desired, the availability of these edges is determined by the required positions and motions.

Next we consider the placement of an edge  $k$  between unspecified nodes  $(i, j)$ . Recall that we write solutions to unspecified node positions and motions as  $\mathbf{v} = W\boldsymbol{\alpha} + \mathbf{v}^*$ , with homogeneous coordinates that must satisfy

$$\begin{bmatrix} \boldsymbol{\alpha}_i^T & 1 \end{bmatrix} \begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_i \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\alpha}_j^T & 1 \end{bmatrix} \begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_j \\ 1 \end{bmatrix} = 0,$$

where  $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2$  are the coordinates for unspecified nodes 1 and 2, respectively. We can rewrite these conditions as

$$\begin{aligned} \boldsymbol{\alpha}_i^T A \boldsymbol{\alpha}_i &= -2B^T \boldsymbol{\alpha}_i - C \\ \boldsymbol{\alpha}_j^T A \boldsymbol{\alpha}_j &= -2B^T \boldsymbol{\alpha}_j - C. \end{aligned}$$

Now we incorporate the constraint placed by the edge

$$\dot{g}_k = (\mathbf{x}_{Ui} - \mathbf{x}_{Uj})^T (\dot{\mathbf{x}}_{Ui} - \dot{\mathbf{x}}_{Uj}) = 0,$$

and we realize that because  $\mathbf{v}_i = W\boldsymbol{\alpha}_i + \mathbf{v}^*$  and  $\mathbf{v}_j = W\boldsymbol{\alpha}_j + \mathbf{v}^*$  have the first  $d$  entries as the unspecified node positions, and the second  $d$  entries as the unspecified node motions,

we can write the vector

$$\mathbf{v}_d = \mathbf{v}_i - \mathbf{v}_j = \begin{bmatrix} \mathbf{x}_{U_i} \\ \dot{\mathbf{x}}_{U_i} \\ c_i \end{bmatrix} - \begin{bmatrix} \mathbf{x}_{U_j} \\ \dot{\mathbf{x}}_{U_j} \\ c_j \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{U_i} - \mathbf{x}_{U_j} \\ \dot{\mathbf{x}}_{U_i} - \dot{\mathbf{x}}_{U_j} \\ c_i - c_j \end{bmatrix} = W(\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_j).$$

Then we can use the same matrix  $O$  used in the derivation of the quadratic constraint

$$O = \frac{1}{2} \begin{bmatrix} 0_{d \times d} & I_{d \times d} & 0_{d \times 1} \\ I_{d \times d} & 0_{d \times d} & 0_{d \times 1} \\ 0_{1 \times d} & 0_{1 \times d} & 0_{1 \times 1} \end{bmatrix},$$

and write our edge constraint as

$$\begin{aligned} \dot{g}_k &= (\mathbf{x}_{U_i} - \mathbf{x}_{U_j})^T (\dot{\mathbf{x}}_{U_i} - \dot{\mathbf{x}}_{U_j}) \\ &= \mathbf{v}_d^T O \mathbf{v}_d \\ &= (\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_j)^T W^T O W (\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_j) \\ &= (\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_j)^T A (\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_j) \\ &= \boldsymbol{\alpha}_i^T A \boldsymbol{\alpha}_i + \boldsymbol{\alpha}_j^T A \boldsymbol{\alpha}_j - 2\boldsymbol{\alpha}_i^T A \boldsymbol{\alpha}_j \\ &= -2B^T \boldsymbol{\alpha}_i - C - 2B^T \boldsymbol{\alpha}_j - C - 2\boldsymbol{\alpha}_i^T A \boldsymbol{\alpha}_j \\ &= -2 \begin{bmatrix} \boldsymbol{\alpha}_i^T & 1 \end{bmatrix} \begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_j \\ 1 \end{bmatrix} \\ &= 0, \end{aligned}$$

such that our selection of  $\boldsymbol{\alpha}_i, \boldsymbol{\alpha}_j$  must satisfy

$$\begin{bmatrix} \boldsymbol{\alpha}_i^T & 1 \end{bmatrix} Q \begin{bmatrix} \boldsymbol{\alpha}_i \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\alpha}_j^T & 1 \end{bmatrix} Q \begin{bmatrix} \boldsymbol{\alpha}_j \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\alpha}_i^T & 1 \end{bmatrix} Q \begin{bmatrix} \boldsymbol{\alpha}_j \\ 1 \end{bmatrix} = 0.$$

To gain an intuition for the implications of this constraint, suppose we fix one unspecified node such that  $\alpha_i^*$  is fixed to satisfy  $\begin{bmatrix} \alpha_i^{*T} & 1 \end{bmatrix} Q \begin{bmatrix} \alpha_i^* \\ 1 \end{bmatrix} = 0$ . Then we can rewrite the edge constraint as

$$\begin{aligned} \dot{g} &= \alpha_i^{*T} A \alpha_i^* + \alpha_j^T A \alpha_j - 2 \alpha_i^{*T} A \alpha_j \\ &= \begin{bmatrix} \alpha_j^T & 1 \end{bmatrix} \underbrace{\begin{bmatrix} A & -A \alpha_i^* \\ -\alpha_i^{*T} A & \alpha_i^{*T} A \alpha_i^* \end{bmatrix}}_{Q^*} \begin{bmatrix} \alpha_j \\ 1 \end{bmatrix} \\ &= 0. \end{aligned}$$

Hence, we see that if we fix the first unspecified node with  $\alpha^*$ , then the second unspecified node must lie at the intersection of two quadratic constraints:  $\begin{bmatrix} \alpha_j^{*T} & 1 \end{bmatrix} Q \begin{bmatrix} \alpha_j^* \\ 1 \end{bmatrix} =$

$$\begin{bmatrix} \alpha_j^{*T} & 1 \end{bmatrix} Q^* \begin{bmatrix} \alpha_j^* \\ 1 \end{bmatrix} = 0.$$

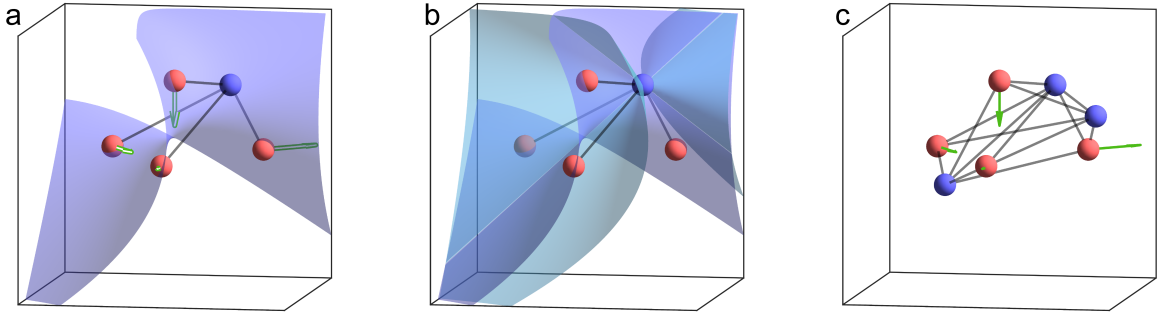


Figure 49: **Constraining Networks with Non-Bipartite Edges.** (a) Illustration of solution space (dark blue curve) of 4 specified nodes (red nodes) with stipulated motions (hollow green arrows), and an unspecified node (blue) on the solution space. (b) The second solution space (light blue curve) satisfying the second conic equation involving  $Q^*$ . (c) A non-bipartite network with no states of self-stress and a  $D = 1$  finitely deformable desired conformational motion (solid green arrows) by placing two unspecified nodes at the intersection of curves.

As an example, we consider  $d = 3$  dimensions for  $n = 4$  specified nodes, and we place one

unspecified node  $\mathbf{x}_{U1} \in \mathcal{M}$  in the solution space (Fig. 49a). Next, we use this unspecified node position to compute the second quadratic matrix  $Q^*$  and the corresponding solution space (Fig. 49b), and place nodes  $\mathbf{x}_{U2}, \mathbf{x}_{U3} \in \mathcal{M} \cap \mathcal{M}^*$  connected to all specified nodes and unspecified node 1. We see that our final network has  $N = 7$  nodes with  $dN = 21$  state variables, and  $E = 14$  edges for a total of  $N_{DOF} = 7$  degrees of freedom, and a  $D = 1$  conformational motion that is achieved with no states of self stress (Fig. 49c).

### 7.10. Network Combinations for Finite Motions

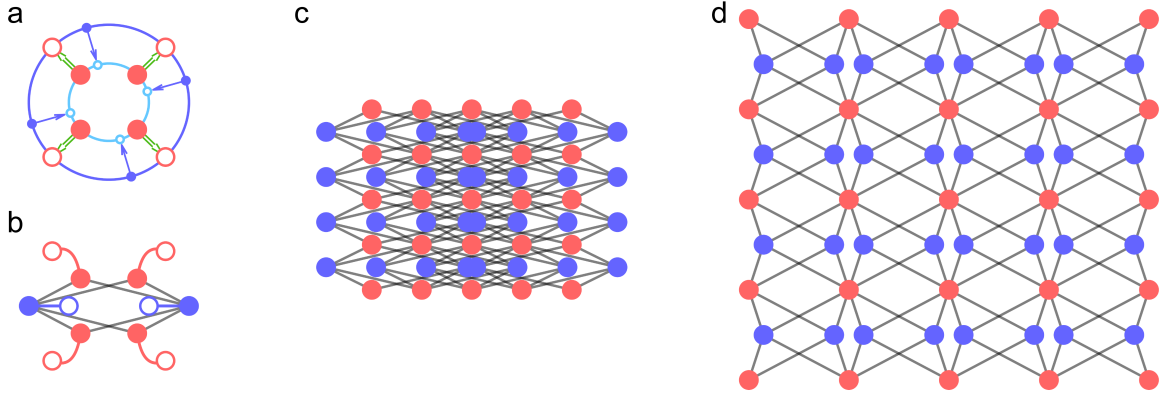


Figure 50: **Combining Networks with Finite Positions.** (a) Illustration of solution spaces for the initial (dark blue) and final (light blue) unspecified node positions that retain edge lengths at the initial (red, filled) and final (red, hollow) specified node positions. (b) Simulated trajectory of a network with  $D = 1$  conformational motions reaching the final position. (c) Initial and (d) expanded networks of combined modules that exactly reach the combined final positions.

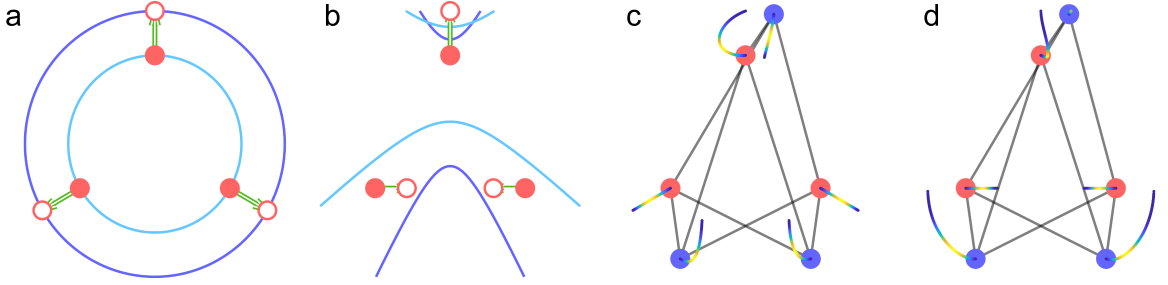
The ability to design networks with specified finite initial and final positions serves even greater utility when considering combining network motions. Similar to the infinitesimal case, we can design modules that preserve symmetries in their finite trajectories, and also reach a desired final position (Fig. 50a–b). Combining these modules into a larger network will then begin at the combined initial positions (Fig. 50c), and exactly reach their combined final positions (Fig. 50d).



### 7.11. Tristable Networks Using Intersections of Finite Motion Solution Spaces

We demonstrate in the main text that we can generate bistable networks by first stipulating some finite initial  $\mathbf{x}_S(0) = \mathbf{x}_S^0$  and final  $\mathbf{x}_S(T) = \mathbf{x}_S^*$  positions, computing a solution space  $\mathcal{M}$ , and placing unspecified nodes until we have  $D = 0$  conformational motions. In this case, the initial and final positions retain edge lengths such that the energy is zero at both. However, because there are no conformational motions, the bonds must bend with some energy cost to transition between them.

Here we start with some initial position  $\mathbf{x}_S^0$ , and we stipulate two finite final positions,  $\mathbf{x}_S^*, \mathbf{x}_S'$ , generating two solution spaces  $\mathcal{M}^*, \mathcal{M}'$  (Fig. 51ab). By placing unspecified nodes at their intersection  $\mathbf{x}_{Uj} \in \mathcal{M}^* \cap \mathcal{M}'$  such that  $D = 0$ , we now have *three* finite positions that have the same edge lengths, and therefore have 0 energy (Fig. 51cd).



**Figure 51: Tristable Networks Using Solution Space Intersections.** (a) Illustration of solution spaces for the initial (dark blue) and final (light blue) unspecified node positions that retain edge lengths at the initial (red, filled) and final (red, hollow) specified node positions for the first stipulated final position  $\mathbf{x}_S^*$ , and (b) for the second stipulated final position  $\mathbf{x}_S'$ . (c) Simulated minimum energy trajectories by setting boundary conditions on the bottom two specified nodes to smoothly transition from initial to final state  $\mathbf{x}_S^*$  (d) and final state  $\mathbf{x}_S'$ . Here, the color of the trajectory represents the potential energy stored in the springs at that point. Note how the initial and both final positions have 0 energy.

## CHAPTER 8 : Nonlinear Dynamics & Chaos in Conformational Changes of Mechanical Metamaterials

### 8.1. Motivation

From cell membrane channels (Li et al., 2015) to medical stents (Mori and Saito, 2005), mechanical systems play crucial roles in the critepp (Patek et al., 2007a; Macol et al., 2001a; Burrows and Sutton, 2013) and engineered (Fu et al., 2016a; Zigoneanu et al., 2014; Surjadi et al., 2019a) world. What makes these systems useful is their ability to change their geometry in a coordinated way to amplify motion or to dramatically change size. Despite their differences, each of these systems can be represented as a mechanical network, where the rigid edges encode constraints due to physical limbs or forces, and the nodes represent joints or constituent elements. A simple and powerful framework for understanding the relationship between network structure and coordinated motion is structural rigidity theory (Crapo, 1979a), originating from early and seminal work by J. C. Maxwell (Maxwell, 1864b; Calladine, 1978; Jacobs and Thorpe, 1995). Here, the difference between the numbers of node coordinates and edges yields the number of coordinated motions.

However, the successful design of coordinated motions depends not only on their existence, but also on the time-evolving network geometry for their duration. The specific geometry is determined by the edge constraints, just as a robot's limbs constrain its configuration. Several works provide design principles relating edge placement to node motions in small networks (Hartenberg and Danavit, 1964; Kim et al., 2019b; McCarthy and Soh, 2010; Connelly and Schlenker, 2010; Stern et al., 2020), and to detailed single-node trajectories or local perturbations in large networks (Rocks et al., 2019; Flechsig, 2017). Other studies explore lattices in the study of topological mechanics in networks (Kane and Lubensky, 2014; Mao and Lubensky, 2018; Sato and Tanaka, 2018; Rocklin et al., 2017) and in origami (Liu et al., 2018; Chen et al., 2016; Melancon et al., 2021), along with sequential and branched motions (Rafsanjani et al., 2019; Coulais et al., 2017; Lubbers and van Hecke, 2019; Stern et al., 2018; Pellegrino, 2001), and they examine design considerations such

as flexible deformation (Bertoldi et al., 2017b), connection topology (Kolken and Zadpoor, 2017; Kolken et al., 2018), and environmental responsivity (Jackson et al., 2018).

Many of these studies take advantage of the simple yet powerful idea to decompose networks into properties of unit cells and their interactions, and to study lattices of — and defects in — identical unit cells. In this work, we build upon this idea to design complex unit cell properties that yield exotic and chaotic behaviors in lattices of identical cells. We further extend these ideas to program arbitrary shape changes and folding sequences in networks by designing and combining *non-identical cells*. Excitingly, many techniques are being developed to physically construct complex networks (Overvelde et al., 2016; Cui et al., 2019; Zhao et al., 2018). As the interest in these systems has grown across many disciplines, it is now timely to develop a general framework for designing specific geometric trajectories in large networks.

Here, we develop such a framework by designing specific properties of shape change in unit cells and their interactions. The manuscript is organized as follows. In Section 8.2, we review mathematical and numerical foundations that we use to study shape changes. In Section 8.3, we formalize principles of how the shape change of a network unit comprising few elements determines the shape change of larger networks comprising many such units. In Section 8.6, we reverse-engineer this process to design units that, when combined, yield targeted global shape change. In Section 8.7, we explore the design space of these units to further design the folding sequence of the network chains. Using these principles, we finally design exotic and nonlinear functions such as a mechanical AND gate in Section 8.8 and chaotic conformational change in Section 8.9, and we construct physical networks in Section 8.11.

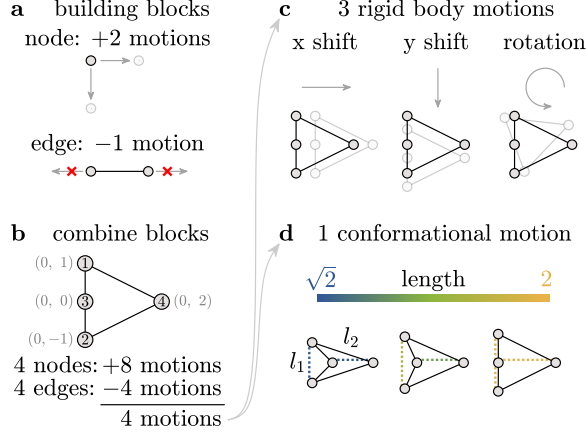


Figure 52: **Constraints & Conformational Motions.** (a) Schematic of a node and an edge embedded in 2 dimensions, where a node adds two motions (one in each dimension), and an edge removes the one motion that changes its length. (b) A network of 4 nodes and 4 edges yields a total of 4 motions, (c) 3 of which are rigid body motions, and (d) 1 of which is a conformational motion.

## 8.2. Mathematical Framework

### 8.2.1. Constraint Counting

Coordinated motions arise from the arrangement of physical forces between constituent elements such as tension and compression transmitted through a rigid robot limb, which we model as distance constraints (edges) between point particles (nodes). In 2-dimensional space, each node  $i$  has two coordinates,  $x_i$  and  $y_i$ , thereby allowing two motions. Each edge  $k$  of length  $d_k$  between nodes  $i$  and  $j$  must keep a constant length

$$(x_i - x_j)^2 + (y_i - y_j)^2 = d_k^2, \quad (8.1)$$

thereby removing one motion (Fig. 52a). Hence, the number of motions in a network without redundant constraints (see Section II C) is given by

$$M = 2N - E, \quad (8.2)$$

where  $N$ ,  $E$ , and  $M$  are the numbers of nodes, edges, and motions, respectively. As such, a network of 4 nodes and 4 edges contains  $8 - 4 = 4$  motions (Fig. 52b). Three motions preserve the distance between all nodes through translations and rotation, and are called *rigid body motions* (Fig. 52c). The fourth motion changes the lengths  $l_1$  and  $l_2$  between unconnected nodes, and is called a *conformational motion* (Fig. 52d)

Throughout, we will use the italicized variable  $d$  to refer to the distance between nodes connected by an edge, the unitalicized symbol  $d$  to refer to the differential operator, and the variable  $l$  to refer to the distance between nodes that are not connected by an edge.

### 8.2.2. Defining the Set of Motions

While we can visually intuit the motions of networks comprising few nodes as in Figure. 52d, we seek a quantitative framework to define such motions for much larger networks. We outline a common framework from rigidity theory (Mao and Lubensky, 2018) that relates *changes* in node coordinates to *changes* in edge lengths. Then the set of allowed node motions are those that cause 0 change in edge length.

For a set of  $N$  nodes  $\mathcal{V} = \{1, \dots, N\}$  connected by  $E$  edges  $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ , any edge  $k$  connecting nodes  $i$  and  $j$  has length  $d_k$  according to Eq. 8.1. To relate changes in node coordinates to changes in edge lengths, we take the derivative of Eq. 8.1 and divide by  $2d_k$  to yield

$$\frac{(x_i - x_j)}{d_k}(\mathrm{d}x_i - \mathrm{d}x_j) + \frac{(y_i - y_j)}{d_k}(\mathrm{d}y_i - \mathrm{d}y_j) = \mathrm{d}d_k. \quad (8.3)$$

We now obtain the desired relationship between node motions,  $\mathrm{d}x$  and  $\mathrm{d}y$ , to changes in edge length,  $\mathrm{d}d_k$ .

Next, we notice that if we know the node positions  $x$  and  $y$  as constants, then the equation is *linear* in the node motion variables:  $\mathrm{d}x$  and  $\mathrm{d}y$ . Due to this linearity, if we write  $\mathbf{x}$  as the  $2N$  dimensional vector of node positions,  $\mathrm{d}\mathbf{x}$  as the  $2N$  dimensional vector of node motions, and  $\mathrm{d}\mathbf{d}$  as the  $E$  dimensional vector of changes in edge length, then we can concisely write

Eq. 8.3 for all edges as

$$fC d\mathbf{x} = d\mathbf{d}. \quad (8.4)$$

Here,  $C = C(\mathbf{x})$  is the *compatibility matrix* of size  $E \times 2N$  comprised predominantly of zeros (Mao and Lubensky, 2018). For every  $k$ -th row in  $C$ , the only non-zero entries are  $(x_i - x_j)/d_k$  multiplying  $dx_i$ ,  $(x_j - x_i)/d_k$  multiplying  $dx_j$ ,  $(y_i - y_j)/d_k$  multiplying  $dy_i$ , and  $(y_j - y_i)/d_k$  multiplying  $dy_j$ . Hence, every row of Eq. 8.4 is precisely Eq. 8.3 for the edge corresponding to that row, and the compatibility matrix maps node motions to bond extensions across the entire network.

Finally, because the edges are rigid, we set  $d\mathbf{d} = \mathbf{0}$  such that all infinitesimal node motions that cause 0 change in edge length must satisfy

$$C d\mathbf{x} = \mathbf{0}. \quad (8.5)$$

As a result, the set of all infinitesimal motions that yield zero change in edge length is given by the nullspace  $\mathcal{N}(C)$ . In our simple 4-node network, the three rigid body motions in Fig. 52c and the one conformational motion in Fig. 52d are all contained in the nullspace of  $C$ . Collectively, these motions are referred to as *zero-modes*.

### 8.2.3. Constraint Counting Revisited

Through the compatibility matrix, we make a more nuanced statement about the number of coordinated motions through the *Calladine Index Theorem* (Mao and Lubensky, 2018). The compatibility matrix maps node motions to bond extensions in Eq. 8.4. Additionally, the equilibrium matrix,  $Q = C^\top$ , maps bond tensions  $\mathbf{t}$  to node forces  $\mathbf{f}$  such that

$$Q\mathbf{t} = \mathbf{f}. \quad (8.6)$$

Here, the nullspace of  $Q$  then represents vectors of bond tensions that cancel out to yield 0 net force at the nodes, and are referred to as *states of self-stress* (SSS) (Mao and Lubensky, 2018). These SSS often arise from over-constraining the network through the addition of redundant bonds, but can also arise from geometric singularities through kinematic bifurcations (Kim et al., 2019b).

The Calladine Index Theorem relates the columnspaces and nullspaces of  $C$  and  $Q$ . From the rank-nullity theorem, we know that a system with  $S$  states of self-stress has the relation

$$\text{rank}(Q) + S = E, \quad (8.7)$$

and that a system embedded in 2 dimensions with  $M$  zero-modes has the relation

$$\text{rank}(C) + M = 2N. \quad (8.8)$$

Because  $\text{rank}(Q) = \text{rank}(C)$ , we substitute to obtain

$$M = 2N - E + S. \quad (8.9)$$

Hence, the number of motions is almost the same as for constraint counting in Eq. 8.2, while accounting for SSS (see Ref. (Mao and Lubensky, 2018) for additional details). Unless stated otherwise, our systems have  $S = 0$ .

#### 8.2.4. Instantiating & Simulating Networks

Now that we have defined the space of allowed node motions, how do we evolve our networks along their conformational motion? Our approach involves four steps, where the ultimate goal is to remove the rigid body motions from the set of all motions to isolate the conformational motion.

First, at simulation step  $k = 0$ , we instantiate our network by choosing the node coordinates

$x_i[0]$  and  $y_i[0]$  for  $i = 1, \dots, N$ , and defining the edge placements between node pairs. Importantly, we note that choosing the node coordinates and edge placements determines the length  $d_k$  of each edge  $k$ . Hence, we are able to fully construct the compatibility matrix  $C$  from the node coordinates and edge placements alone.

Second, at simulation step  $k$  starting at  $k = 0$ , we collect all of the node positions into a  $2N$  dimensional vector  $\mathbf{x}[k]$ , construct our compatibility matrix  $C[k] = C(\mathbf{x}[k])$ , and compute the set of allowed node motions through the nullspace  $\mathcal{N}(C[k])$ . We collect the basis set that spans the nullspace as a  $2N \times M$  matrix,  $P[k]$ .

Third, we define the basis set of rigid body motions  $R[k] = [\mathbf{x}_x, \mathbf{x}_y, \mathbf{x}_{rot}[k]]$  and quotient them out of our set of allowed motions to yield the conformational motion. Numerically, we can implement this quotient by taking the nullspace of the projection of  $R[k]$  onto  $P[k]$  as  $\mathcal{N}(R[k]^\top P[k])$ , collecting the basis vectors that span this nullspace into a  $M \times M - 3$  matrix  $Q$ , and projecting the nullspace back into the coordinate space as

$$d\mathbf{x}[k] = P[k]Q. \quad (8.10)$$

Unless stated otherwise,  $d\mathbf{x}[k]$  will always be a vector, because our systems will always have  $M = 4$  motions, such that removing the 3 rigid body motions will leave behind 1 conformational motion. We normalize the conformational motion such that  $d\hat{\mathbf{x}}[k] = d\mathbf{x}[k]/\|\mathbf{x}[k]\|_2$ .

Finally, we evolve the network forward by numerically integrating the differential in Eq. 8.10. Specifically, at each time step  $k$ , we evolve the node positions forward from  $\mathbf{x}[k]$  to  $\mathbf{x}[k+1]$  using a 10-th order Runge-Kutta scheme that relies on the evaluation of steps 2 and 3 at each substep ((Feagin, 2007)). The reason for such a high order integration scheme will become clear in the results, as high accuracy of numerical integration is necessary for simulating networks with chaotic behaviors.



### 8.2.5. Motivating Statement & Outline

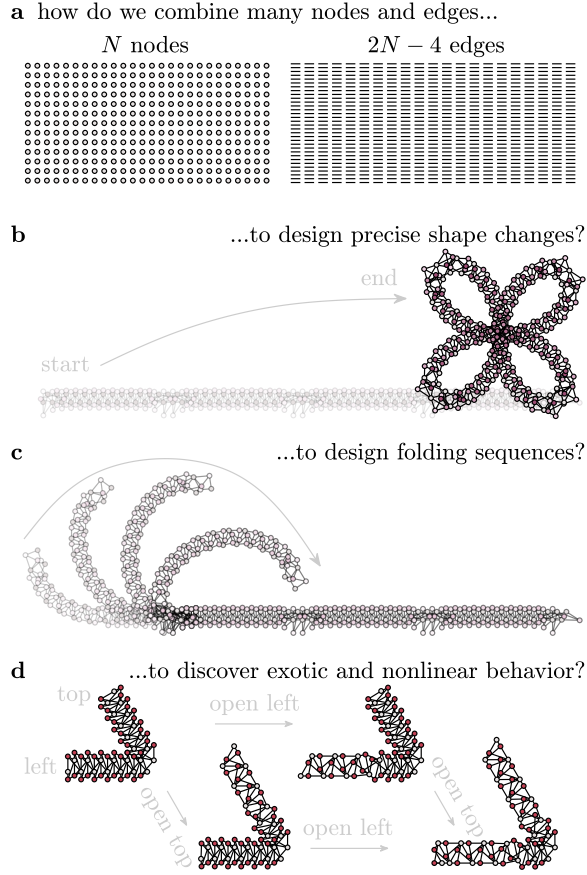


Figure 53: **Motivation for the Results.** (a) A set of 338 nodes and 672 edges that, through our results, can be designed to (b) have precise shape changes and (c) folding sequences. (d) A network that has been designed to behave as a mechanical AND gate.

Conformational motions endow networks with functions that depend on targeted changes in shape. The design of a specific shape change is determined by the node positions and edge placements, and is made difficult by the nonlinearity of the constraints, even in networks of few nodes (Eq. 8.1). Given the vast design space in systems of many nodes and edges (Fig. 53a), what are the organizational principles that enable us to design precise shape changes (Fig. 53b), folding sequences (Fig. 53c), and exotic and nonlinear behavior (Fig. 53d)?

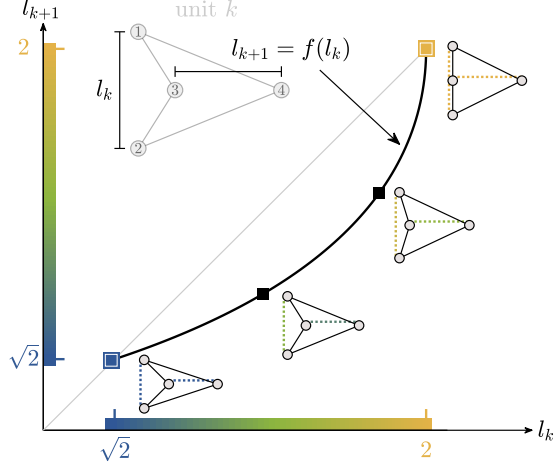


Figure 54: **Conformational motion as a map.** Plot of the lengths  $l_k$  and  $l_{k+1}$  between unconnected nodes in the example unit (Fig. 52) as it changes shape.

### 8.3. A 4-Bar Linkage Example

To understand the principles that govern shape change in networks of many elements, we first develop intuition for shape change in units comprising a few elements. Specifically, we study the shape change of the 4-bar linkage previously shown in Figure 52. We observe that the length between unconnected nodes, namely length  $l_k$  between nodes 1 and 2, and  $l_{k+1}$  between nodes 3 and 4, change throughout the motion (Fig. 54). If we plot these two lengths along the motion, we obtain a curve that maps length  $l_k$  to length  $l_{k+1}$  as a function

$$l_{k+1} = f(l_k). \quad (8.11)$$

The equation of this specific unit's map is  $f(l_k) = \sqrt{5 - l_k^2/4} - \sqrt{1 - l_k^2/4}$ , and is derived via the edge constraints given in Eq. 8.1.

#### 8.3.1. Combining Units Acts as a Map Iteration

This map immediately motivates a simple and powerful way to construct a network of many nodes whose shape change is fully known. Specifically, if we could somehow combine these units such that the lengths of subsequent units are functions of the lengths of previous

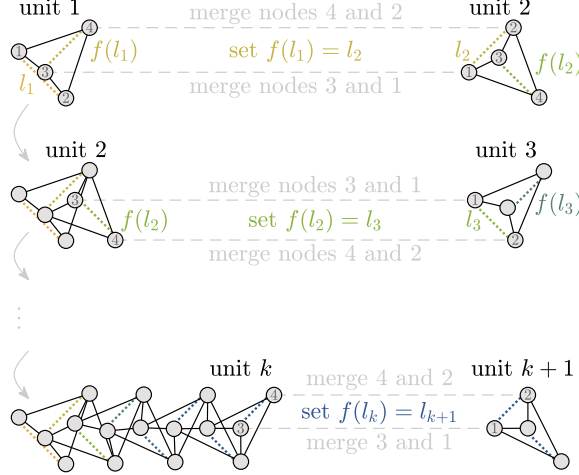


Figure 55: **Combine Units by Merging Nodes.** A unit  $k + 1$  is combined to unit  $k$  by merging nodes. First the length  $l_{k+1}$  between nodes 1 and 2 of unit  $k + 1$  is set equal to the length  $f(l_k)$  between nodes 3 and 4 of unit  $k$ . Then, node 2 of unit  $k + 1$  is *merged* to node 4 of unit  $k$  by overlapping and gluing the nodes such that they become the same node, and node 1 of unit  $k + 1$  is merged to node 3 of unit  $k$  in the same manner.

units, then we could write the conformation of all units as a function of the first, such that  $l_{k+1} = f^k(l_1)$ . Through this relation, we could simply and explicitly parameterize the shape of the entire network through a single parameter.

To achieve this relationship, consider two 4-bar linkage units, 1 and 2. Unit 1 has length  $l_1$  between nodes 1 and 2, and length  $f(l_1)$  between nodes 3 and 4. Unit 2 has length  $l_2$  between nodes 1 and 2, and length  $f(l_2)$  between nodes 3 and 4 (Fig. 55). If we set  $l_2$  of unit 2 equal to  $f(l_1)$  of unit 1, then we can combine units 1 and 2 by *merging* nodes, by which we mean overlapping and gluing node 3 of unit 1 and node 1 of unit 2 such that they become the same node, and by overlapping and gluing node 4 of unit 1 and node 2 of unit 2 in the same way. Thus, the lengths of unit 2 are determined by those of unit 1, such that  $f(l_2) = f(f(l_1))$  (Fig. 55, top).

Afterwards, we add another unit — unit 3 — whose nodes 1 and 2 define length  $l_3$ , and whose nodes 3 and 4 define length  $f(l_3)$ . By setting  $l_3$  of unit 3 equal to length  $f(l_2)$  of unit 2, we can combine units 2 and 3 by merging node 3 of unit 2 with node 1 of unit 3 such that they become the same node, and by merging node 4 of unit 2 with node 2 of unit 3

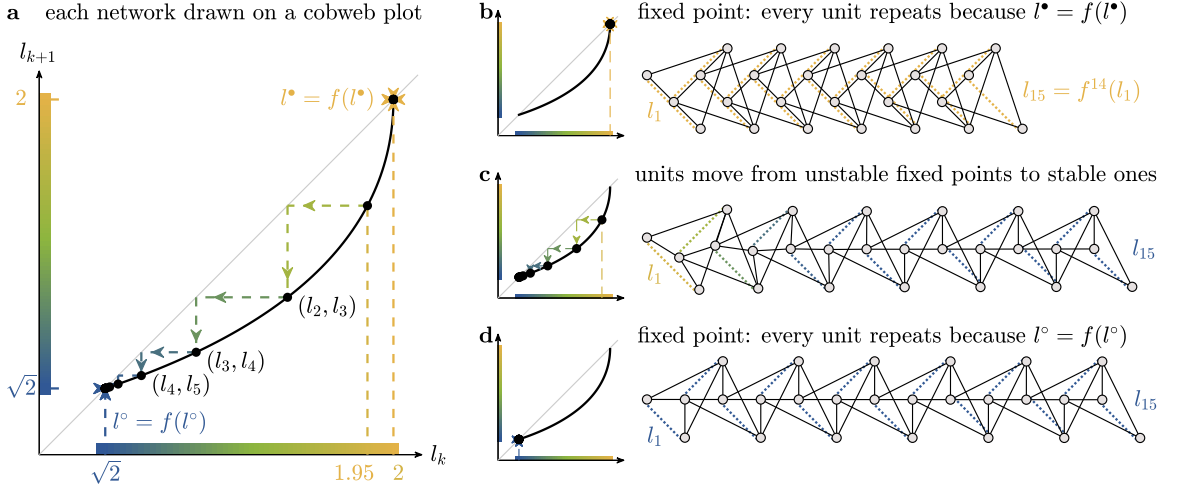
in the same way. Thus, the shape of unit 3 is also determined by that of unit 1, such that  $f(l_3) = f(f(l_2)) = f(f(f(l_1)))$  (Fig. 55, middle).

We can continue this process as often as we like, such that unit  $k$  combines to unit  $k + 1$  by first setting length  $l_{k+1}$  of unit  $k + 1$  equal to  $f(l_k)$  of unit  $k$ , and then merging node 3 of unit  $k$  with node 1 of unit  $k + 1$ , and node 4 of unit  $k$  with node 2 of unit  $k + 1$  (see the Appendix in Chapter 9). In this way, the shape of unit  $k + 1$  is determined by that of unit 1, such that

$$l_{k+1} = f^k(l_1), \quad (8.12)$$

as we show in the bottom of Figure 55. This equation is referred to as an *iterated map* in nonlinear dynamics.

### 8.3.2. Visualizing Map Iteration as a Cobweb Plot



**Figure 56: Shape & Folding Sequence of Iterated Maps.** (a) A plot of the curve  $l_{k+1} = f(l_k)$  for the 4-bar linkage example, with three cobweb plots drawn at initial lengths of  $l_1 = 2, 1.95$ , and  $\sqrt{2}$ . (b) At an initial length of  $l_1 = 2$ , all subsequent units  $k$  also have the identical shape  $l_k = 2$  because  $f(2) = 2$ . (c) At an initial length of  $\sqrt{2} < l_1 = 1.95 < 2$ , the subsequent units are no longer identical, and converge towards the stable fixed point at  $l^o = \sqrt{2}$ . (d) At an initial length of  $l_1 = \sqrt{2}$ , all subsequent units  $k$  also have the identical shape  $l_k = \sqrt{2}$ .

To develop an intuition for the relationship between the map iteration and the network chain's geometry, we can visually represent the map iteration as a *cobweb plot* (Zhou et al., 2017), which consists of horizontal and vertical lines in the plot of  $l_{k+1} = f(l_k)$ . Such visualizations will show us the properties of the map (Eq. 8.12) that are useful to design.

To draw a cobweb plot, we start by drawing the map between lengths  $l_{k+1} = f(l_k)$  (Fig. 56a). A cobweb plot begins at the initial length  $l_1$  along the horizontal axis, and we draw a vertical line up to the function  $f$  until it reaches the ordered pair  $(l_1, f(l_1))$ . In our example, one of the cobweb plots begins at  $l_1 = 1.95$ , and has a vertical line drawn to the ordered pair  $(1.95, f(1.95))$ . Next, a horizontal line is drawn to the line  $l_{k+1} = l_k$  to reach the ordered pair  $(f(l_1), f(l_1))$  to prepare the coordinates for the next function evaluation. This process of drawing a vertical line to ordered pair  $(l_k, f(l_k))$ , followed by a horizontal line to the diagonal  $l_k = l_{k+1}$ , is repeated for as many units as are in the network chain.

As a result, each ordered pair  $(l_k, f(l_k))$  represents the conformation of unit  $k$ , and the entire cobweb plot represents the conformation of the entire network chain at a particular initial length  $l_1$ . In Figure 56a, we show three cobweb plots corresponding to three network chains, where  $l_1$  of unit 1 begins at  $l_1 = 2$ ,  $l_1 = 1.95$ , and  $l_1 = \sqrt{2}$  (Fig. 56a), with the corresponding network conformations shown in Figure 56b, c, and d, respectively. Importantly, we note that these three networks are identical in terms of bond lengths and connectivity. They only differ in the initial length  $l_1$ . Additionally, we note that the network can continuously deform its geometry from Figure 56b to Figure 56d along one conformational motion, without changing bond lengths or connection topology.

We highlight two key observations from these cobweb plots. The first is that there are some points where all units are identical, namely  $l = \sqrt{2}$  and  $l = 2$  (Fig. 56b,d). This property is in some sense ideal, because we can know simply and precisely the conformation of every unit in the network. The second observation is that if the initial length is in between these points such that  $\sqrt{2} < l_1 < 2$ , then the units seem to converge to  $\sqrt{2}$  (Fig. 56c). This property clues us in to how we can design the folding sequence of the network.

#### 8.4. Network Conformation is Known at Fixed Points

While it is true that the conformation of every unit  $k$  is determined by Equation 8.12, the continued analytical or numerical evaluation of the map  $f$  to determine length  $l_{k+1} = f^k(l_1)$  is quite cumbersome. However, there are special lengths  $l^*$  known as *fixed points* that map back to themselves such that

$$l^* = f(l^*), \quad (8.13)$$

where the conformation of every unit is easily known. This is because if  $f(l^*) = l^*$ , then  $f^k(l^*) = l^*$ , and every unit is in the same conformation.

In our 4-bar linkage example, these fixed points are significant because every unit takes on an identical, repeating conformation, which we will refer to as a *periodic state*. We show the network in the  $l^\bullet = 2$  periodic state in Figure 56b, and in the  $l^\circ = \sqrt{2}$  periodic state in Figure 56d. In between these two states is an intermediary conformation when  $\sqrt{2} < l_1 < 2$ , as shown in Figure 56c. Here, the network is still constructed from the same 4-bar linkage unit with the same bond lengths and connection topology as the previous and subsequent units. We refer to this construction as one having a periodic structure. However, because the length of any unit  $k$  does not repeat across the network, it does not have a periodic state.

The motivation and significance for studying these fixed points is that at the fixed points, *the conformation of every unit is simply and completely known*. Hence, in Section 8.6, we will fulfill our first aim, to design precise shape changes in networks of many elements (Fig. 53b), by designing units that adopt precise geometries at a common fixed point.

#### 8.5. Folding Sequence is Determined by Stability

In addition to the conformation of the network chain, the iterated map can also tell us about the change in conformation, or the folding sequence, of the network. This change is

simply understood by taking the derivative of Equation 8.11 to yield the slope

$$s_k = \frac{dl_{k+1}}{dl_k} = f'(l_k). \quad (8.14)$$

Intuitively, for any unit  $k$ , the slope simply tells us whether a perturbation in length  $l_k$  yields a larger or smaller perturbation in length  $l_{k+1}$ . If the slope  $|s_k| < 1$ , then the magnitude of perturbation decreases, and the map at the point  $l_k$  is said to be *stable*. If the slope  $|s_k| > 1$ , then the magnitude of perturbation increases, and the map at  $l_k$  is said to be *unstable*. If  $|s_k| = 1$ , the magnitude of perturbation remains the same, and the map is said to be *marginally stable*. When considering this same change in the iterated map equation for the entire network chain, we obtain

$$s = \frac{dl_{k+1}}{dl_1} = \prod_{i=1}^{k+1} s_i, \quad (8.15)$$

which tells us whether a perturbation in  $l_1$  will be larger or smaller than a perturbation in  $l_{k+1}$ .

If  $|s| > 1$ , the perturbation in  $l_{k+1}$  will be larger than that at  $l_1$ , and the network will begin changing conformation at the  $l_{k+1}$  end. In our specific example, the fixed point  $l^\bullet = 2$  is unstable, because the slope of the map has magnitude greater than 1. Hence, in Equation 8.15, we observe that  $|dl_{k+1}| > dl_1$ , and expect the network at the fixed point  $l^\bullet = 2$  in Figure 56b to begin folding from the  $l_{k+1}$  end, which is true (Fig. 56c).

If  $|s| < 1$ , the perturbation in  $l_{k+1}$  will be smaller than that in  $l_1$ , and the network will begin changing conformation at the  $l_1$  end. In our specific example, the fixed point  $l^\circ = \sqrt{2}$  is stable, because the slope of the map has magnitude less than 1. Hence, in Equation 8.15, we observe that  $|dl_{k+1}| < dl_1$ , and expect the network at the fixed point  $l^\circ = \sqrt{2}$  in Figure 56d to begin folding from the  $l_1$  end, which is true (Fig. 56c).

The motivation and significance of studying the stability is that the *folding sequence of the network is determined by the stability of the unit maps*. Hence, in Section 8.7, we will fulfill

our second aim, to design folding sequences (Fig. 53c), by designing units that are stable or unstable at a particular geometry.

## 8.6. Designing Network Shape

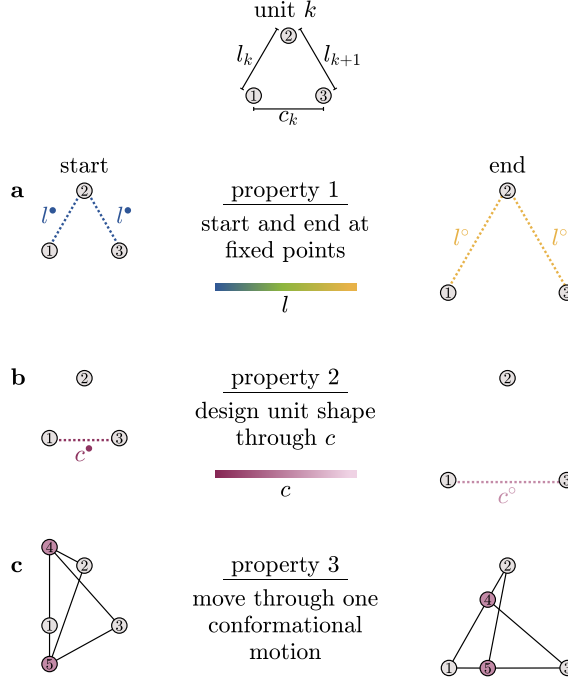


Figure 57: **Properties of Unit Design.** Drawing of a unit with three nodes, 1, 2, and 3, with length  $l_k$  between nodes 1 and 2, length  $l_{k+1}$  between nodes 2 and 3, and length  $c_k$  between nodes 1 and 3. We seek to design units (a) that start and end at fixed points where  $l_k = l_{k+1} = l^\bullet$  and  $l_k = l_{k+1} = l^\circ$  are valid conformations, (b) whose start and end geometry can be programmed to start at  $c^\bullet$  and end at  $c^\circ$ , and (c) can transition from the start to the end shape with one conformational motion.

Using the fact that we know the conformation of all units when at a fixed point, we seek to achieve our first aim to design the shape of the network chain when the units are at the fixed points (Fig. 53b). Recalling our previous 4-bar linkage example, we immediately encounter a problem: identical units have identical geometries at fixed points, such that the network chain forms a straight line (Fig. 56b,d). Thus, we are motivated to design our own, non-identical units that share fixed points but differ in their precise geometry.



### 8.6.1. Motivating the Unit Design Procedure

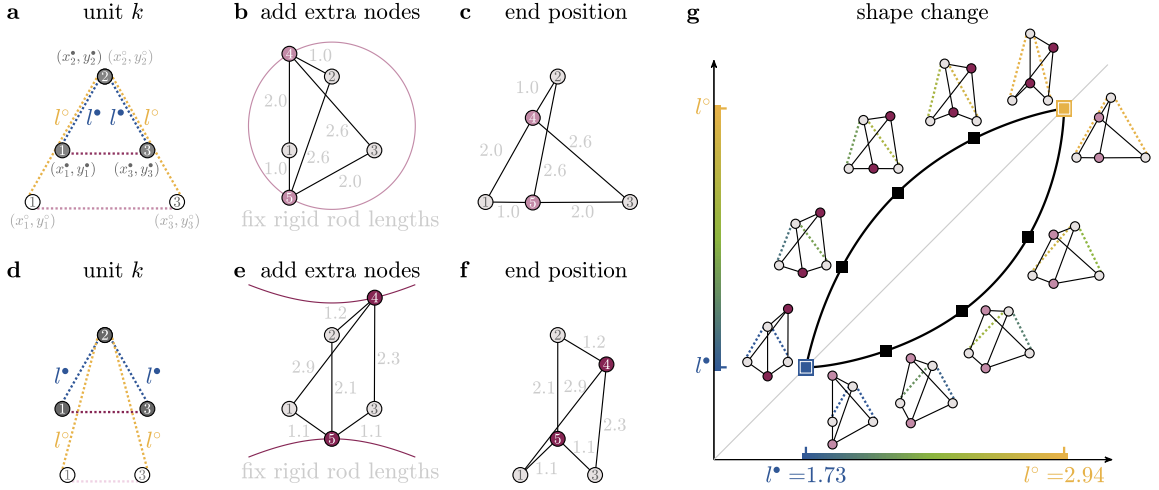
To achieve desired shape changes in the network chain, we require that the composite units satisfy three key properties. We will approach these properties in a constructive manner, beginning with a unit comprising a set of 3 nodes — 1, 2, and 3 —, and defining length  $l_k$  between nodes 1 and 2, length  $l_{k+1}$  between nodes 2 and 3, and length  $c_k$  between nodes 1 and 3 (Fig. 57). We begin with three nodes because that is the smallest number of nodes whereby we can define three independent lengths,  $l_k$ ,  $l_{k+1}$ , and  $c_k$ , to achieve three desired properties of unit design that yield desired shape changes. We describe these three properties one at a time.

The first property is that, among this set of nodes, there exist two conformations (node coordinate positions) where the lengths between unconnected nodes,  $l_k$  and  $l_{k+1}$ , are equal. We refer to the length at the first such conformation as the *start* fixed point,  $l^\bullet$ , and the length at the second such conformation as the *end* fixed point,  $l^\circ$  (Fig. 57a). This property ensures that, even if we construct and combine non-identical units, they will all share identical start and end fixed points. This sharing means that if one unit is in a conformation where a length is at a fixed point such that  $l_k = l_{k+1} = l^\bullet$  or  $l_k = l_{k+1} = l^\circ$ , then all units, despite being nonidentical, also exist in a conformation where their lengths are at  $l^\bullet$  or  $l^\circ$ , respectively. Importantly, we note that because fixed points require the lengths  $l_k$  and  $l_{k+1}$  to be identical such that  $l_k = l_{k+1}$ , the three nodes will form an isosceles triangle at the start coordinates (with isosceles side  $l^\bullet$ ) and end coordinates (with isosceles side  $l^\circ$ ).

The second property differentiates our designed unit from the example 4-bar linkage unit, such that we want a design parameter that changes the *shape* of the unit, and thereby the network, at the fixed points. Hence, the second property is that we have a parameter  $c$  through which we can design the shape of the unit at the start fixed point as  $c^\bullet$  and at the end fixed point as  $c^\circ$  (Fig. 57b).

The third and final property is that the unit achieves properties 1 and 2 through one conformational motion. This property allows us to write the lengths between unconnected nodes of a unit as a map (Eq. 8.11), thereby allowing us to write the lengths across the entire network as an iteration of this map, precisely as in Figure 55. Because adding edges between nodes 1, 2, and 3 would interfere with desired properties 1 and 2, we will constrain our unit to have 1 conformational motion by adding additional nodes 4 and 5 that are fully connected to nodes 1, 2, and 3 (Fig. 57c). The resulting network will have 5 nodes corresponding to 10 state variables, and 6 edges corresponding to 6 constraints, leaving us with  $10 - 6 = 4$  total motions, and thereby 3 rigid body motions and 1 conformational motion.

### 8.6.2. The Unit Design Procedure



**Figure 58: Designing Unit Geometry at Fixed Points.** (a) Node coordinates that start at a fixed point  $l^\bullet$  and end at another fixed point  $l^\circ$ , where the length  $c$  is larger at the end than at the start such that  $c^\circ > c^\bullet$ . (b) The maroon curve is the *solution space*. By placing extra nodes 4 and 5 on the solution space and fully connecting them to nodes 1, 2, and 3 with rigid bonds, (c) there exists an end position of nodes 4 and 5 that keeps all bond lengths the same. (d) Another unit with the same start fixed point  $l^\bullet$  and end fixed point  $l^\circ$ , but the length  $c$  is chosen to be smaller at the end than at the start such that  $c^\circ < c^\bullet$ . (e) By placing extra nodes 4 and 5 on the solution space and connecting them to nodes 1, 2, and 3 with rigid edges, (f) there exists an end position that retains all bond lengths. (g) Plots of the conformational motion of both units as they change shape from the start fixed point to the end fixed point.

Now that we have motivated the desired properties of our unit, we will achieve these properties using a method from prior work (Kim et al., 2019b). We notice that our base unit forms a triangle with nodes 1, 2, and 3 as the vertices, and that properties 1 and 2 specify the lengths of the triangle edges at the start and end conformations (Fig. 57). By specifying the lengths of all triangle edges, we also specify all node coordinates up to isometric transformations (i.e. translation, rotation, mirror images). Hence, in our unit, designing a unit whose lengths between node pairs are fixed at the start and end conformation is equivalent to designing a unit whose node coordinates are fixed at the start and end conformation.

Hence, we begin our unit design procedure by defining  $(x_i^\bullet, y_i^\bullet)$  as the start coordinate of node  $i$ , and by defining  $(x_i^\circ, y_i^\circ)$  as the end coordinate of node  $i$  (Fig. 58a). First, we use trigonometry to convert the start edge lengths,  $l^\bullet, l^\bullet, c^\bullet$ , into start node coordinates,  $(x_1^\bullet, y_1^\bullet), (x_2^\bullet, y_2^\bullet), (x_3^\bullet, y_3^\bullet)$ , and the end edge lengths,  $l^\circ, l^\circ, c^\circ$ , into end node coordinates,  $(x_1^\circ, y_1^\circ), (x_2^\circ, y_2^\circ), (x_3^\circ, y_3^\circ)$ . In this particular example, we choose  $l^\bullet = \sqrt{3}$  to be the length of the start fixed point, and  $l^\circ = 1.7 \cdot \sqrt{3}$  to be the length of the end fixed point. We also choose  $c^\bullet = \sqrt{3}$  to be the start shape parameter, and vary the end shape parameter,  $c^\circ$ , across units to change the units' shape. These specific lengths were chosen for the purpose of demonstration, and that the method does not require these particular lengths.

Now that we have chosen the coordinates for nodes  $i \in \{1, 2, 3\}$  at the start configuration,  $(x_i^\bullet, y_i^\bullet)$ , and end configuration,  $(x_i^\circ, y_i^\circ)$ , that satisfy properties 1 and 2, we must decide on the position of the two extra nodes  $j \in \{4, 5\}$  while ensuring that the edges have the same lengths at the start and end configurations. Otherwise, the edges cannot be rigid. This condition is enforced by first setting the squared length of the edges at the start coordinates equal to the squared length of the edges at the end coordinates as

$$(x_i^\bullet - x_j^\bullet)^2 + (y_i^\bullet - y_j^\bullet)^2 = (x_i^\circ - x_j^\circ)^2 + (y_i^\circ - y_j^\circ)^2, \quad (8.16)$$

where  $(x_i^\bullet, y_i^\bullet)$  and  $(x_i^\circ, y_i^\circ)$  are fixed constants (Fig. 59a,d), and then solving for the start and end positions of the extra nodes,  $(x_j^\bullet, y_j^\bullet)$  and  $(x_j^\circ, y_j^\circ)$ . The solutions of  $(x_j^\bullet, y_j^\bullet)$  and

$(x_j^\circ, y_j^\circ)$  for Eq. 8.16 then define the start and end positions of the added node  $j$  that do not change the length of any edges, allowing them to be rigid (see the Appendix in Chapter 9 for details on solving the equations and on how the node placement fixes the rod lengths).

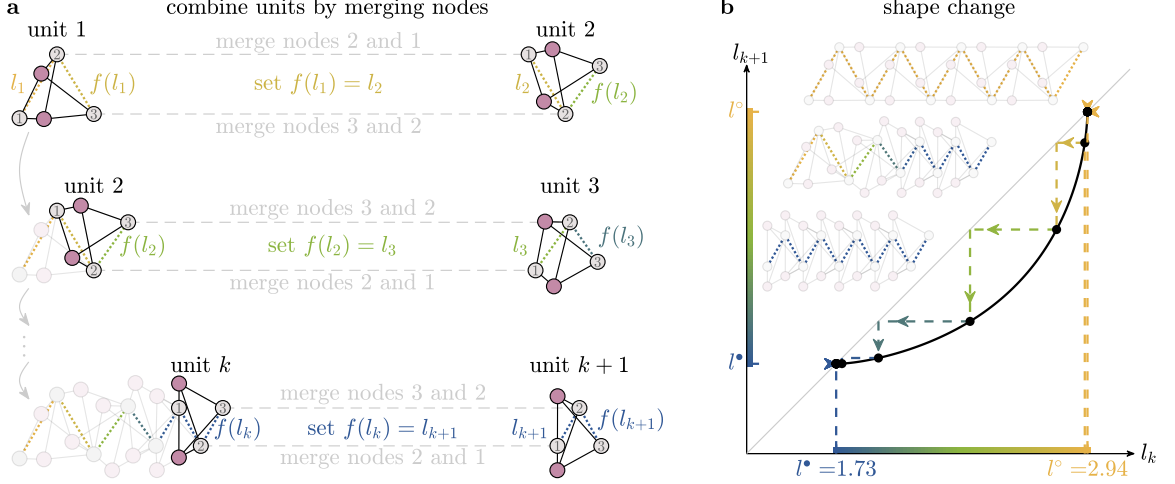


Figure 59: **Representing the Combining of Designed Units as an Iterated Map.**

(a) Combining units by merging nodes. A unit  $k + 1$  is combined to unit  $k$  by merging nodes. First, the length  $l_{k+1}$  between nodes 1 and 2 of unit  $k + 1$  is set equal to length  $f(l_k)$  between nodes 2 and 3 of unit  $k$ . Then, node 2 of unit  $k + 1$  is overlapped and glued together with node 3 of unit  $k$  to become a single node, and node 1 of unit  $k + 1$  is overlapped and glued together with node 2 of unit  $k$ . (b) A cobweb plot of the curve  $l_{k+1} = f(l_k)$  for the designed unit, with three cobweb plots drawn at initial lengths of  $l_1 \approx 1.73$ ,  $2.93$ , and  $2.94$ , with a drawing of the network corresponding to each cobweb plot.

The solutions to Eq. 8.16 have a very particular structure. At the start position, it feels intuitive that we should be able to place our extra node  $j$  at any location  $(x_j^\bullet, y_j^\bullet)$ , and connect it with edges of length  $d_k$  that are equal to the distances between nodes  $i$  and  $j$ . However, when we then move nodes  $i \in \{1, 2, 3\}$  to their desired end position  $(x_i^\circ, y_i^\circ)$ , we find that there are typically no end positions for the extra node  $(x_j^\circ, y_j^\circ)$  that keep the edge lengths the same at  $d_k$ . Intuitively, this lack of solution arises from the fact that while there are 3 added edges from nodes  $i \in \{1, 2, 3\}$  to node  $j$  (and thereby 3 constraints to satisfy from Eq. 8.16), there are only 4 variables for the extra node positions  $(x_j^\bullet, y_j^\bullet, x_j^\circ, y_j^\circ)$ . Hence, there is typically only a one-dimensional solution for the extra node positions, which is defined by a conic section due to the constraints being quadratic (Kim et al., 2019b). We

call the set of start positions  $(x_j^\bullet, y_j^\bullet)$  satisfying Eq. 8.16 the *solution space* (Fig. 58b,e). By placing our two nodes  $j \in \{4, 5\}$  on the solution space (Fig. 58b), our unit reaches the desired final position (Fig. 58c), and does so along 1 conformational motion (Fig. 58g).

This procedure can now be used to design non-identical units with the same fixed points  $l^\bullet$  and  $l^\circ$ , but a different end geometry given by the shape parameter  $c^\circ$ . In contrast to the first unit that we designed where the end shape parameter was larger than the start shape parameter ( $c^\circ > c^\bullet$ , Fig. 58a-c), we can design a second unit where the end shape parameter is smaller than the start shape parameter ( $c^\circ < c^\bullet$ , Fig. 58d-f). Because the two examples have different end positions, solving Eq. 8.16 yields different solution spaces (Fig. 58b *versus* Fig. 58d). By adding nodes  $j \in \{4, 5\}$  on the solution space for the first unit (Fig. 58b), the unit has the same bond lengths between the first start and end positions (Fig. 58c). If we place extra nodes  $j \in \{4, 5\}$  on the solution space for the second unit (Fig. 58d), then the unit has the same bond lengths between the second start and end positions (Fig. 58f).

This method is not specific to only the two examples show in Figure. 58. For any start  $(x_i^\bullet, y_i^\bullet)$  and end  $(x_i^\circ, y_i^\circ)$  positions, we can solve Eq. 8.16 for the solution space, place extra nodes  $j \in \{4, 5\}$  along the solution space, connect all nodes  $j$  to all nodes  $i$ , and guarantee that the bond lengths at the start and end positions are equal. The solution space has two important implications. First, if we were to place the extra nodes  $j$  *outside* of the solution space, then we are guaranteed that our nodes will not go from the start positions  $(x_i^\bullet, y_i^\bullet)$  to the end positions  $(x_i^\circ, y_i^\circ)$ . This is because Eq. 8.16 defines *all* placements of node  $j$  that preserve edge length. By placing node  $j$  outside of the solution space, we are guaranteed to fail at finding an end position of node  $j$  that maintains edge length. The second implication is that *different* desired start and end positions for nodes  $i$  define *different* solution spaces. Between examples 1 and 2, the different end positions  $(x_i^\circ, y_i^\circ)$  generated different solution spaces (Fig. 58b *versus* Fig. 58e). This is because the constant parameters of Eq. 8.16 changed, thereby changing the form of the solution space. Varying  $c^\circ$  generates a range of solution spaces from which we construct our units.

### 8.6.3. Combining Designed Units With Map Iteration

Now that we have designed units that move from a desired start fixed point  $l^\bullet$  to a desired end fixed point  $l^\circ$  (property 1) and from a desired start shape  $c^\bullet$  to a desired end shape  $c^\circ$  (property 2) along one conformational motion (property 3), we need to test whether we can construct network chains as in Figure 56. To do so, we take our first designed unit from Figure 58b, and combine many such units in a manner similar to our 4-bar linkage example, whereby we write the conformation of the  $k$ -th unit as repeated functions of the starting length  $l_1$ .

As in the 4-bar linkage example in Section 8.3.1, we will combine units by merging nodes. We begin with our designed unit 1, with length  $l_1$  between nodes 1 and 2, and length  $f(l_1)$  between nodes 2 and 3 (Fig. 59a). We then take a second unit with length  $l_2$  between nodes 1 and 2, and length  $f(l_2)$  between nodes 2 and 3, and set  $l_2 = f(l_1)$ . Finally, we combine units 1 and 2 by merging node 2 of unit 1 with node 1 of unit 2, and by merging node 3 of unit 1 with node 2 of unit 2. In Figure 59a, the nodes to be merged are marked with dashed gray lines. By “merge,” we again mean that we move unit 2 over to unit 1 and overlap the nodes to be merged (e.g., node 2 of unit 1 has the same spatial coordinates as node 1 of unit 2), and glue them together such that the overlapped nodes become the same node. In this manner, we can write the conformation of unit 2 as given by  $f(l_2)$  as a function of  $l_1$  through  $f(l_2) = f(f(l_1))$ .

To continue the process of combining units, we add another unit, unit 3, with length  $l_3$  between nodes 1 and 2, and length  $f(l_3)$  between nodes 2 and 3. As before, we set length  $l_3$  of unit 3 equal to length  $f(l_2)$  of unit 2, and combine units 2 and 3 by merging node 2 of unit 2 with node 1 of unit 3, and node 3 of unit 2 with node 2 of unit 3 (Fig. 59a). In this manner, we can write the conformation of unit 3 as given by  $f(l_3)$  as a function of  $l_1$  through  $f(l_3) = f(f(l_2)) = f(f(f(l_1)))$ .

To continue the process of combining units more generally, we add unit  $k + 1$  with length

$l_{k+1}$  between nodes 1 and 2, and length  $f(l_{k+1})$  between nodes 2 and 3. As before, we set length  $l_{k+1}$  of unit  $k+1$  equal to length  $f(l_k)$  of unit  $k$ , and combine units  $k$  and  $k+1$  by merging node 2 of unit  $k$  with node 1 of unit  $k+1$ , and by merging node 3 of unit  $k$  with node 2 of unit  $k+1$  (Fig. 59a, see the Appendix in Chapter 9 for extra details on how units are combined). In this manner, we can write the conformation of unit  $k+1$  as given by  $f(l_{k+1})$  as a function of  $l_1$  through the iterated map  $f(l_{k+1}) = f^k(l_1)$  (Eq. 8.12).

Hence, all of the intuitions that we derived regarding the 4-bar linkage units in Section 8.3 translate directly to our own designed units. Specifically, the intuitions that units can be combined such that the shape of unit  $k+1$  can be written as the iterated map  $l_{k+1} = f(l_k)$  in Section 8.3.1, that the network's conformational change can be visualized according to a cobweb plot in Section 8.3.2, that units have identical *states* at fixed points in Section 8.4, and that the network's folding sequence is determined by the stability of the units in Section 8.5, all translate directly to our designed units (Fig. 59).

#### 8.6.4. *Motivating the Network Design Procedure*

Now that we can design general shape changes in units, how can we select the specific units that will yield networks with desired global shape? For example, how do we design non-identical units that combine to form a network chain that folds into a complex shape such as a quadrifolium (Fig. 60a)? To reverse-engineer this process, we can reverse the order of this question to ask: how can we decompose a desired global shape into a network comprising specific unit shapes?

To answer this question, first recall from Section 8.4 that at a fixed point, all unit geometries are known because the lengths  $l_k$  and  $f(l_k)$  are equal to either the start fixed point  $l^\bullet$  or the end fixed point  $l^\circ$ . Second, recall from Section 8.6.2 that we can choose the start and end values of the shape parameter,  $c^\bullet$  and  $c^\circ$  respectively, at the start and end fixed points (Fig. 58). By choosing units whose shape parameters at the end fixed point follow along a trace (Fig. 60a), we can design the global shape of the network chain.

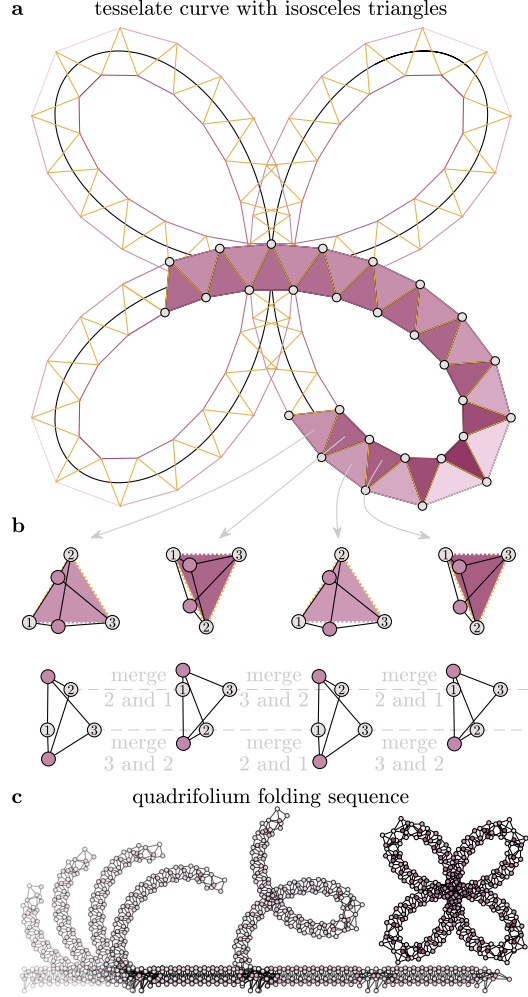


Figure 60: **Designing Precise Network Geometry.** (a) To design a network chain that forms a desired curve, we tessellate the curve with isosceles triangles, where each triangle represents a unit. The gold edges represent  $l_k$  and  $l_{k+1}$ , while the purple edges represent  $c$ . (b) The units are then constructed by placing extra nodes along the solution space defined by the start and end positions (Section 8.6.2). Then the units are combined into a chain by merging nodes (Section 8.6.3). (c) The resulting network consists of non-identical units that have different bond lengths, and thereby the network has non-periodic *structure*. However, because all units were designed to share the same start fixed point  $l^\bullet$  and end fixed point  $l^\circ$ , the network starts at a periodic *state* at  $l_k = l^\bullet$ , and undergoes one conformational motion to form the desired curve at the second periodic *state*  $l_k = l^\circ$ .

#### 8.6.5. Selecting Units That Yield Global Network Shape

Hence, we seek for the end node positions of all units to trace our desired shape at the end fixed point, which we accomplish by tessellating the end global shape with the end node



positions,  $(x_i^\circ, y_i^\circ)$  while enforcing that these positions are at the end fixed point  $l^\circ$ . To demonstrate this process, we will construct a network that folds into a quadrifolium as the desired final shape (Fig. 60a, black). The specific equation of the trace is given by

$$r = a \sin(2\theta), \quad 0 \leq \theta < 2\pi, \quad (8.17)$$

where  $r$  and  $\theta$  represent the radial and angular coordinate of the curve, and we use  $a = 16.2$  in our example.

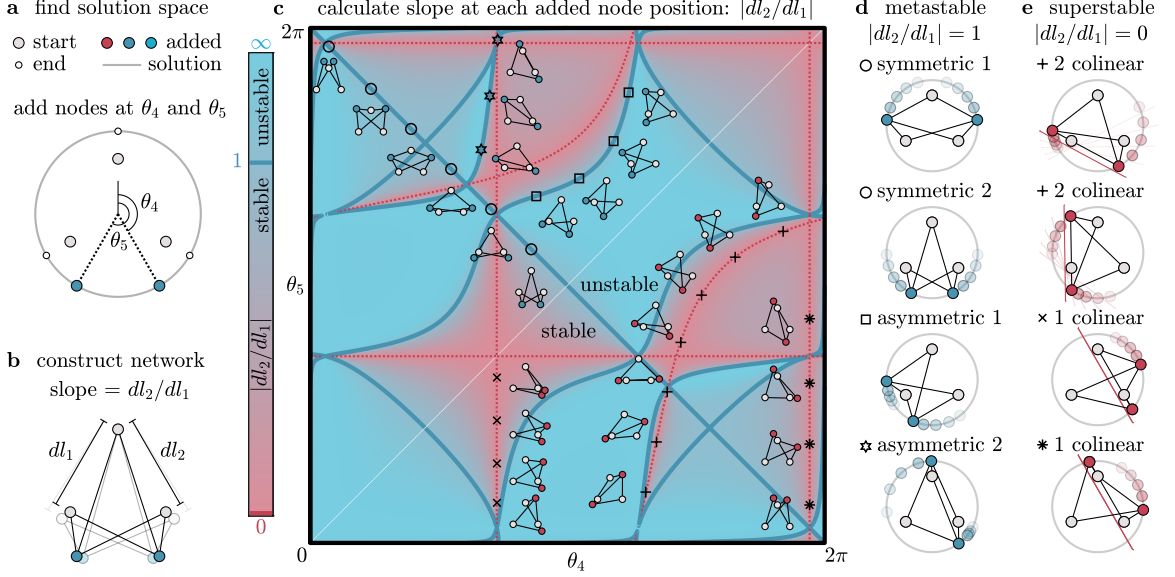
To convert this curve into a network, we tessellate the curve with node coordinates  $(x_i^\circ, y_i^\circ)$  (Fig. 60a). In this figure, each triangle corresponds to the final shape of one unit, where the corners are the final node positions  $(x_i^\circ, y_i^\circ)$ , the isosceles sides are  $l_k = l_{k+1} = l^\circ$  (gold), and the non-isosceles side is the shape variable  $c^\circ$  (purple). The reason why each unit forms an isosceles triangle with isosceles edges between nodes 1 and 2, and between nodes 2 and 3, is because at a fixed point, the length  $l_k$  between nodes 1 and 2, and the length  $l_{k+1}$  between nodes 2 and 3, are equal to the fixed point length  $l_k = l_{k+1} = l^\circ$ . The reason why the units are merged along the isosceles edge is because the units combine by merging nodes that define  $l_k$  and  $l_{k+1}$ , which at the fixed point, is the isosceles edge.

Now that we have the desired end coordinates for nodes 1, 2, and 3 of each unit, we construct each unit according to Section 8.6.2 by placing two nodes  $j \in \{4, 5\}$  satisfying Eq. 8.16, and combine units precisely according to Section 8.6.3 by merging the nodes corresponding to the shared corners between neighboring triangles to form a network (Fig. 60b). At the start fixed point  $l_k = l^\bullet$ , the network begins as a line because all lengths  $c = l^\bullet$ . At the end fixed point  $l_k = l^\circ$ , the shape variables reach their programmed length  $c = c^\circ$ , thereby forming the quadrifolium (Fig. 60c).

In sum, we successfully achieve our first goal, to understand the organizational principles that enable us to design precise shape changes (Fig. 53b) by designing units with one conformational motion that transition between two fixed points with tunable shape (Section 8.6.2),

and by choosing and combining non-identical units whose end shapes trace out the desired shape (Section 8.6.5).

## 8.7. Designing the Conformational Sequence Using Stability



**Figure 61: Designing the Sequence of Conformational Change.** (a) Schematic of the addition of two nodes along the solution space, parameterized by angles  $\theta_4$  and  $\theta_5$ . (b) At the start position, connecting the added nodes to the initial nodes with edges yields one conformational motion, characterized by a slope of  $dl_2/dl_1$ . (c) Phase diagram of the slope magnitude at all placements of added nodes. The solid blue line marks the transition between stable and unstable. The red dashed line marks where the slope = 0. (d) Units with  $|slope| = 1$ , and (e) with  $|slope| = 0$ .

Now that we have the principles for constructing complex shape changes, we move on to our second aim to design a network's folding sequence by designing the stability of the maps of its component units (Fig. 53c). Recall from Section 8.3 that the folding sequence of a network depends on the stability of the maps of the component units. If all units are in a conformation that is stable (i.e.  $|dl_{k+1}/dl_k| < 1$ ), then a perturbation at  $l_1$  decays across units, and the network begins folding at the  $l_1$  end (Fig. 56d). Alternatively, if all units are in a conformation that is unstable, then a perturbation at  $l_1$  grows across units, and the network begins folding at the  $l_{k+1}$  end (Fig. 56b). How can we tune the stability of our units' maps to design the folding sequence?

### 8.7.1. Motivating Stability Design

To design a unit's stability, we must first keep in mind that there already exist constraints from designing a unit's shape in Section 8.6. In Section 8.6.2, we already designed our unit comprising three nodes,  $i \in \{1, 2, 3\}$ , to successfully transition from desired start positions  $(x_i^\bullet, y_i^\bullet)$  to the desired end positions  $(x_i^\circ, y_i^\circ)$  along one conformational motion (Fig. 57). To do this, we added two additional nodes,  $j \in \{4, 5\}$ , and fully connected them to the first three nodes for a total of 6 edges  $\mathcal{E} = \{1, 2, 3\} \times \{4, 5\}$  to yield 1 conformational motion (Fig. 57c).

We found that these additional nodes could not be placed arbitrarily in space. Once we fix the start positions of the added nodes,  $(x_j^\bullet, y_j^\bullet)$ , we also fix the edge lengths, and most choices of edge lengths cannot remain constant at the start and end positions according to Equation 8.16. Instead, in Section 8.6.2, we found that the start positions of each of the added nodes must lie on a 1-dimensional conic section, the *solution space*, for the subsequently fixed edge lengths to remain rigid at the start and end positions (Fig. 58).

An immediate question that arises is precisely *where* on the solution space should nodes  $j \in \{4, 5\}$  be placed? If each node  $j$  can be placed anywhere along the 1-dimensional solution space, then for the two added nodes, we are left with a 2-dimensional parameter space (1-dimensional solution space per node) along which we can add nodes  $j \in \{4, 5\}$ . In addition to the unit shape, can we use these two dimensions to design the stability of the unit at these shapes?

### 8.7.2. Searching the Parameter Space

The positions of the added nodes determine the stability of a unit, which in turn determines the sequence of the full network's shape change. To design a unit's stability, we establish general principles of node placement through the detailed study of one unit whose initial and final node positions generate a circular solution space, along which we add node 4 at  $\theta_4$  and node 5 at  $\theta_5$  (Fig. 61a). At each position  $0 \leq \theta_4, \theta_5 < 2\pi$ , we connect the added nodes

$j \in \{4, 5\}$  to all of the original nodes  $i \in \{1, 2, 3\}$  and compute the slope of length changes as a function of the node positions (Fig. 61b,c).

We observe consistent patterns of node positions for units that are maximally stable (*superstable* (Lee, 2009)) where  $|dl_2/dl_1| = 0$ . Similar to a stable configuration where a perturbation in length  $l_1$  decays as it propagates to length  $l_2$ , a superstable configuration means that a perturbation in length  $l_1$  yields no response to linear order in length  $l_2$ . Hence, a network comprising units in superstable configurations will fully localize their shape change at the  $l_1$  end to linear order. We observe that superstable units entail that both added nodes are co-linear with the node  $i = 1$  that exclusively defines  $l_1$ , or that at least one added node is colinear with both nodes  $i = 2, 3$  defining  $l_2$  (Fig. 61e). The former condition guarantees that the sole motion of node 1 (perpendicular to the co-linearity) is a conformational motion, such that  $|dl_1| \geq 0$  while  $|dl_2| = 0$ . The latter condition guarantees that the motion of nodes 2 and 3 is perpendicular to the direction of their length, such that  $|dl_2| = 0$  while  $|dl_1| \geq 0$ . Hence, we ensure stable units in the quadrifolium by placing nodes near the first co-linear condition (Fig. 60).

We also observe consistent patterns of node positions for units at the transition between stable and unstable (*marginally stable*) where  $|dl_2/dl_1| = 1$  (Fig. 61d). Marginally stable units encompass all symmetric node positions  $\theta_4 = -\theta_5$ , whereby the positions of the added nodes are mirrored across the vertical axis. Additionally, marginally stable units consist of more complex asymmetric node positions (Fig. 61d).

In sum, we successfully achieve our goal of designing not only the shape (Fig. 53b), but also the folding sequence (Fig. 53c) of a network chain, by designing the stability of its component units. We find that stable units can be designed through the co-linear placement of added nodes 4 and 5 (Fig. 61e), while marginally stable units can be designed through the symmetric placement of added nodes (Fig. 61d). Importantly, these principles generalize to solution spaces that are not circles, and we used these principles to choose the stability of the units in Figure 58, and in the quadrifolium Figure 60.

## 8.8. Superstability & the Mechanical AND Gate

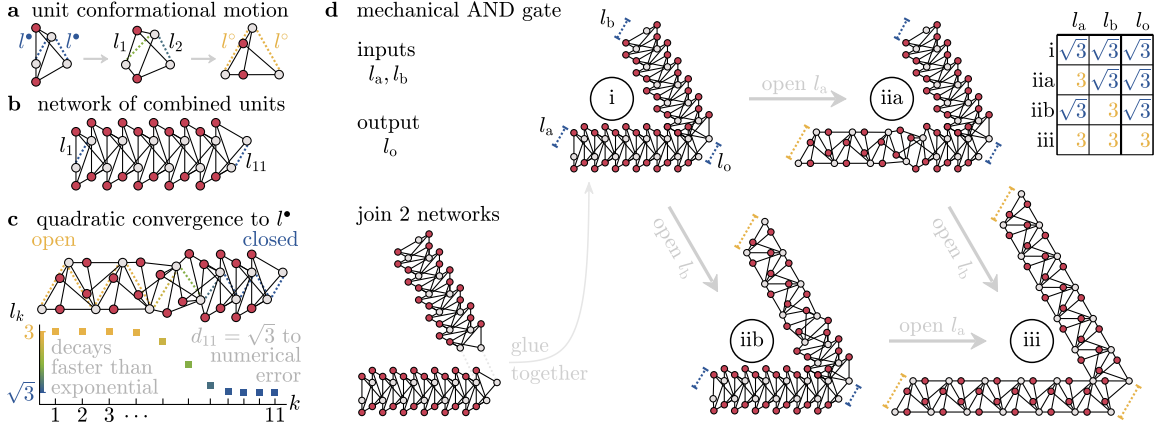


Figure 62: **Superstability & Extreme Localization Through Faster-Than-Exponential Convergence.** (a) Conformational motion of a unit from the superstable fixed point  $l^\bullet$  to the unstable fixed point  $l^\circ$ . (b) Copies of this unit combine to form a network chain that (c) converges to  $l^\bullet$  at a faster-than-exponential rate. (d) Two of these network chains,  $a$  and  $b$ , combine to form a mechanical AND gate, where the inputs  $l_a$  and  $l_b$  are the  $l_1$  ends of each network. From  $i$ , the gate has two independently deformable conformational motions to form any linear combination of  $ii$ a and  $ii$ b. The signal can only continue to  $l_o$  when both inputs are open at  $l_a = l_b = 3$ .

### 8.8.1. Motivating Superstable Convergence

Now that we have achieved the design of network shape and folding sequence (Fig. 53b,c), we move on to the discovery of exotic and nonlinear behavior (Fig. 53d). We have already encountered one such inherently nonlinear phenomenon through superstable unit conformations (Fig. 61e), where a change in length  $l_1$  yields no change in length  $l_2$  to linear order. This means that a network comprising units at a superstable fixed point will converge to that fixed point at a faster than linear rate. What implications does superstability have for our ability to design networks with nonlinear behavior?

### 8.8.2. Utilizing Superstable Convergence

Networks at a superstable fixed point demonstrate a qualitatively more extreme localization of shape change than those at merely stable fixed points. To formalize this concept, we take

the second-order Taylor series expansion of a unit's map,  $l_{k+1} = f(l_k)$ , about a fixed point  $l^*$  with slope  $s = f'(l^*)$  and half of the curvature  $t = f''(l^*)/2$  to yield

$$\Delta l_{k+1} \approx s\Delta l_k + t\Delta l_k^2. \quad (8.18)$$

A network at a stable fixed point (e.g.,  $s = 0.1$ ) converges *linearly* because the quadratic term  $\Delta l_k^2 \ll \Delta l_k$  becomes negligibly small, such that an infinitesimal perturbation  $dl_1$  cannot be registered to numerical precision ( $10^{-16}$ ) after unit  $k = 16$ . In contrast, a network at a superstable fixed point ( $s = 0$ , Fig. 62a,b) converges *quadratically* because the linear term vanishes, such that an infinitesimal perturbation  $dl_1$  cannot be registered in unit  $k = 2$ . Even a finite displacement  $\Delta l_1 = 0.1$  propagates to  $\Delta l_{k+1}$  that is smaller than  $10^{-16}$  after unit  $k = 4$ ,  $10^{-32}$  after unit  $k = 5$ , and  $10^{-64}$  after unit  $k = 6$  (Fig. 62c). As a reference, the ratio of diameters between a classical electron and the observable universe is around  $10^{-42}$ .

This severe localization of shape change renders the  $l_{k+1}$  end effectively rigid, thereby allowing us to design networks with unexpected and nonlinear functions. Here, we demonstrate a mechanical instantiation of an AND gate, which is a binary operator with one Boolean output that depends on two independent Boolean inputs. The Boolean states are the unit's two fixed points,  $l^\bullet = \sqrt{3}$  and  $l^\circ = 3$  (Fig. 62a), and each of the gate's inputs is the length  $l_1$  of a network (Fig. 62b). We combine the two networks by merging the indicated nodes at and near length  $l_{11}$  to form our mechanical AND gate (Fig. 62d,d-i, see the Appendix in Chapter 9 for a physical network).

By constraint counting from Eq. 8.2, the AND gate should have 1 finitely deformable conformational motion, even if it exists at a kinematic bifurcation allowing for 2 infinitesimal motions (Mao and Lubensky, 2018). However, due to the quadratic convergence to the fixed point  $l^\bullet$ , the AND gate begins at a geometry that effectively has 2 independent and finite motions (Fig. 62d-iiia,d-iib), and ends at a geometry that only has one such motion at  $l_{11}$  (Fig. 62d-iii). Hence, the AND gate can finitely access all four combinations of Boolean

inputs, while only one allows for the propagation of the mechanical signal.

## 8.9. Period Doubling Route to Mechanical Chaos

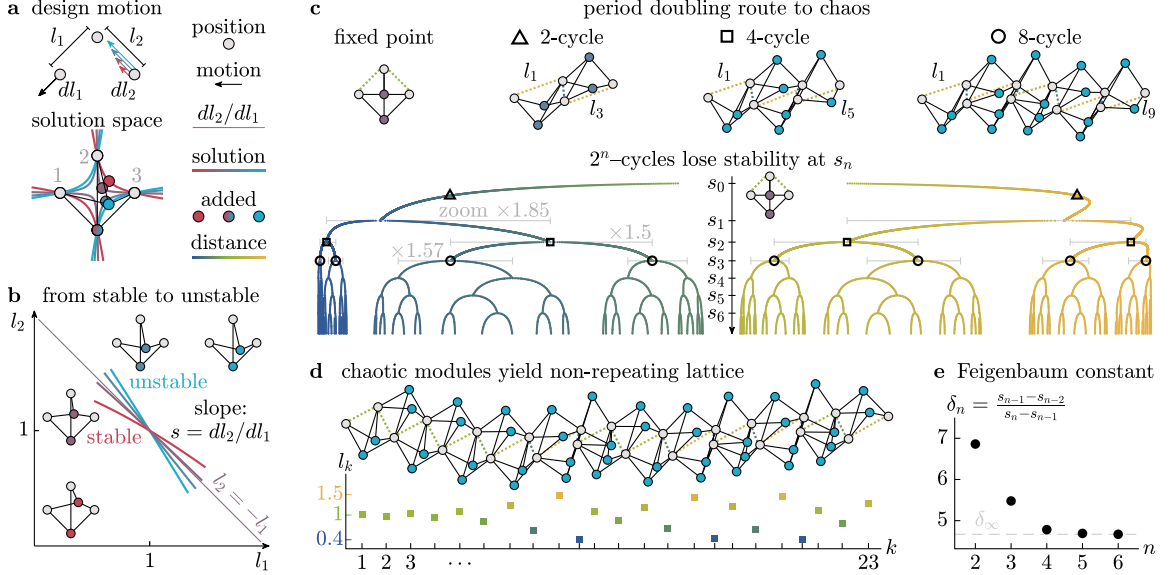


Figure 63: **Mechanical Chaos.** (a) Schematic of the stability design process at the fixed point  $l_1 = l_2 = 1$ , with the designed instantaneous node motions in black and colored arrows, the corresponding solution spaces in colored lines, and the positions of the added nodes for each unit in colored nodes. (b) Slope plots of the conformational motion for each unit. (c) At a slope of  $s_0 = -1$  (left), the fixed point  $l^* = 1$  is marginally stable. As the slope passes  $s_1$ , the fixed point becomes unstable, and a new stable 2-cycle is born (triangle). As the slope continues past  $s_n$ , each  $2^n$  cycle becomes unstable and gives birth to a stable  $2^{n+1}$  cycle, until (d) the unit becomes chaotic with no stable cycles. (e) The ratio of slopes at the bifurcations converges to the Feigenbaum constant. The slope values in this example are  $(s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7) \approx (-1, -1.522545, -1.574527, -1.582104, -1.583487, -1.583777, -1.583838, -1.583851)$ .

### 8.9.1. Motivating Chaotic Divergence

At the opposite extreme, networks comprising units that lose their stability undergo divergent shape changes that are unpredictable and chaotic. Until now, every unstable fixed point has been accompanied by a stable one, to which each subsequent unit  $k+1$  eventually converged. If there were no accompanying stable fixed point, to where would the units converge?

### 8.9.2. Designing Chaotic Divergence

To answer this question, we design the slope of a unit at a fixed point by drawing on prior work (Kim et al., 2019b) to constrain the motion of unconnected nodes  $i \in \{1, 2, 3\}$ . We define  $l_1$  and  $l_2$  to be the length between node pairs  $\{1, 2\}$  and  $\{2, 3\}$ , respectively, and place the nodes at  $(x_i, y_i)$  such that the lengths equal a fixed point  $l_1 = l_2 = 1$  (Fig. 63a). We then choose the instantaneous node motions  $(dx_i, dy_i)$  to achieve a desired slope  $s = dl_2/dl_1$ , where we fix  $dl_1 = 1$  as constant, and vary  $dl_2$ . To achieve  $(dx_i, dy_i)$  as the sole conformational motion, we connect all nodes  $i$  to an added node  $j$ , and solve for the positions  $(x_j, y_j)$  and motions  $(dx_j, dy_j)$  that keep all edge lengths constant by satisfying the derivative of Eq. 8.1

$$(x_i - x_j)(dx_i - dx_j) + (y_i - y_j)(dy_i - dy_j) = 0. \quad (8.19)$$

We call these node positions  $(x_j, y_j)$  the *solution space*, along which we add two nodes  $j \in \{4, 5\}$  to yield a network with 1 conformational motion that achieves the designed slope at the fixed point (Fig. 63b, see the Appendix in Chapter 9 for the design algorithm, the analytical form of the iterated map, and the conditions for a unit conformational motion to act as a map, and Ref. (Kim et al., 2019b) for additional details).

For a unit designed with a stable fixed point ( $|s| < 1$ ), subsequent units converge to the fixed point and assume the same shape (Fig. 63b). As we design units with more negative slopes  $s < -1$ , the fixed point undergoes a bifurcation and becomes *unstable*, giving birth to a stable 2-cycle where every 2nd unit in the network repeats (Fig. 63, triangle). As we design units with increasingly negative slopes, the 2-cycle becomes unstable at slope  $s = s_1$ , giving birth to a stable 4-cycle (Fig. 63, square), which then becomes unstable at  $s = s_2$  and gives birth to a stable 8-cycle (Fig. 63, circle). Continuing this process, each  $2^n$ -cycle loses stability at  $s_n$ , and gives birth to a stable  $2^{n+1}$ -cycle, until the network loses all stable cycles and becomes *chaotic* (Fig. 63d). This process is known as a *period-doubling*



*bifurcation*, and is characterized by the *Feigenbaum constant* (Strogatz, 2018)

$$\delta_\infty = \lim_{n \rightarrow \infty} \frac{s_{n-1} - s_{n-2}}{s_n - s_{n-1}} \approx 4.669, \quad (8.20)$$

to which our units converge (Fig. 63e). Additionally, the chaotic evolution of our units is captured by the *Lyapunov exponent* that quantifies the rate of divergence of subsequent units from infinitesimally nearby initial units, and is given by

$$\lambda(l_1) = \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{k=1}^n \ln |f'(l_k)|. \quad (8.21)$$

We estimate the exponent by averaging across long trajectories from many initial conditions to be 0.246 (see the Appendix in Chapter 9 for the calculation of the exponent), demonstrating positive divergence: a hallmark of chaos.

## 8.10. Period Three Implies Mechanical Chaos

### 8.10.1. Motivating 3-cycle Units

In the previous section, as we changed the edge lengths of a unit to lose stability at a fixed point, the unit underwent a period-doubling route to chaos. While the presence of many-period cycles may be useful for designing metamaterial lattices, each choice of edge length only corresponded to a specific  $2^n$ -cycle. Can a single super-unit with fixed edge lengths yield arbitrarily many cycles?

### 8.10.2. A 3-cycle Unit and Sharkovskii's Theorem

To obtain such a super-unit, all we require is for the unit's map to display a 3-cycle. This requirement is a direct result of Sharkovskii's theorem, which states that for any real interval  $I \subset \mathbb{R}$ , if a map  $f : I \rightarrow I$  has a point of period 3, then it contains a point of period  $k$  where  $k$  is a positive integer (Sharkovskii, 1995). This deceptively simple statement leads to powerful consequences, as a unit whose map contains a 3-cycle not only implies chaos (Li

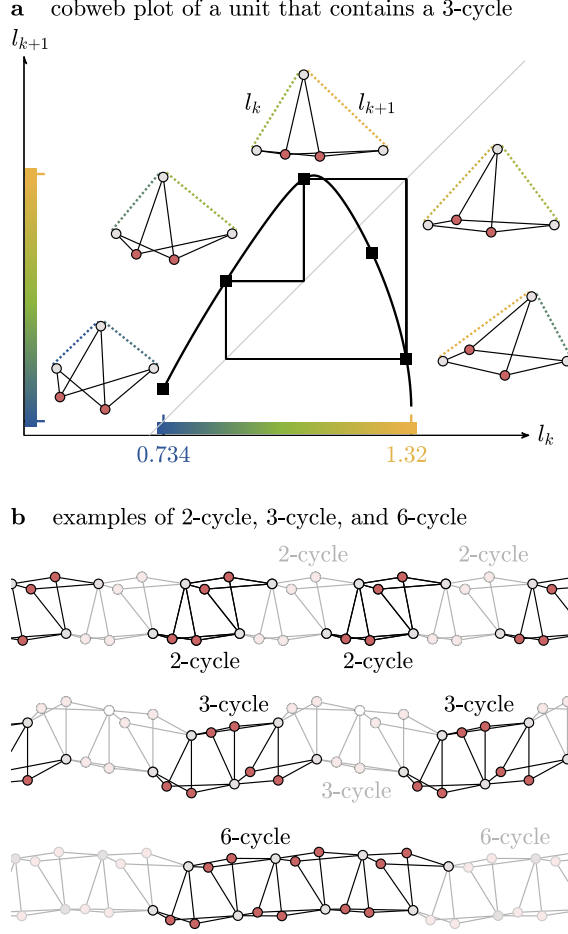


Figure 64: **A 3-Cycle Unit.** (a) Cobweb plot of a unit containing a 3-cycle. (b) Examples of 2-cycle, 3-cycle, and 6-cycle conformations that can be found in this unit.

and Yorke, 2004), but also implies that it can change its shape to yield *any integer-period cycle*.

We discover such a 3-cycle unit (Fig. 64a), and also demonstrate the presence of other positive-integer cycles such as 2-cycles and 6-cycles (Fig. 64b). Importantly, unlike the unit in the period-doubling route to chaos, this unit contains cycles of *all* positive integer periods with *one* single set of edge weights.

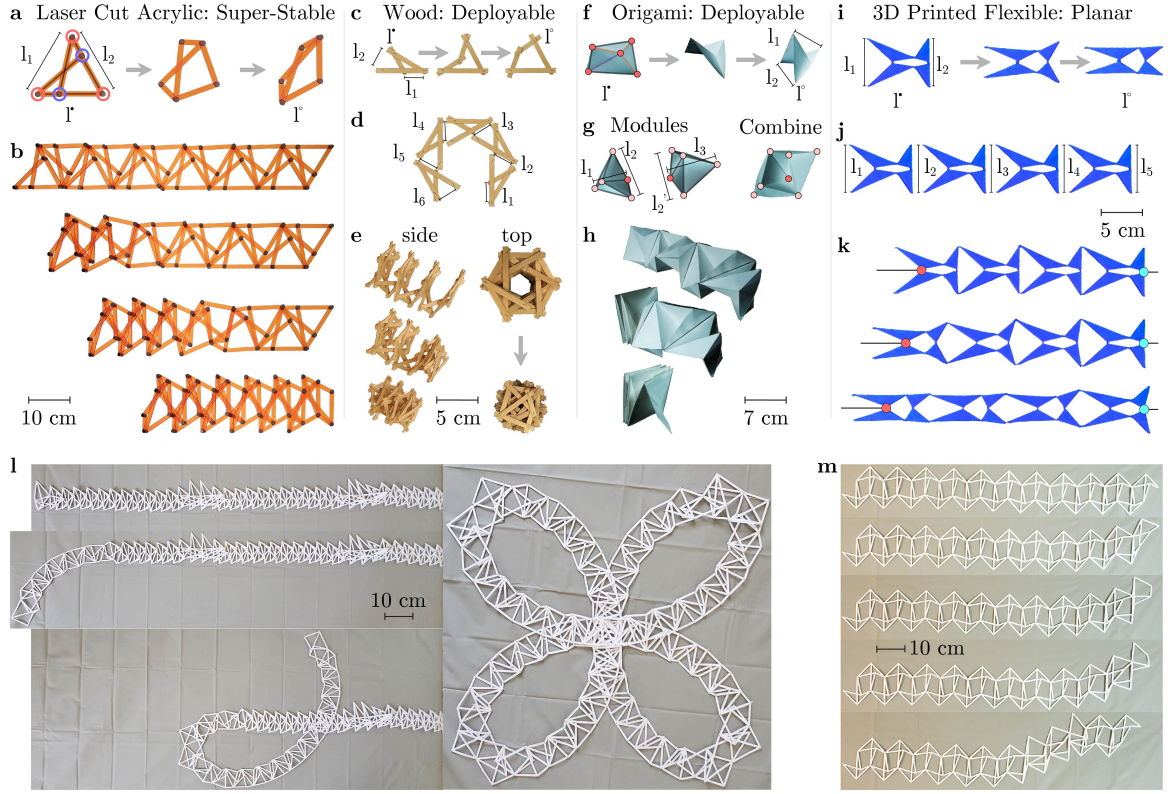


Figure 65: **Physical Construction of Networks.** (a) Photo of a super-stable unit constructed from laser-cut acrylic bars held together by Chicago screws at the joints, transitioning between two fixed points  $l^\bullet$  and  $l^\circ$ . (b) Photos of a combined network collapsing from  $l^\bullet$  to  $l^\circ$ . (c) A 4-bar linkage with two fixed points  $l^\bullet$  and  $l^\circ$ , (d) combined hexagonally into (e) an initially wide spiral helix with a channel  $l^\bullet$ , collapsing sequentially to a narrow closed helix. (f) Photo of a creased square sheet of paper modeled as a linkage with 1 conformational motion moving between two fixed points  $l^\bullet$  and  $l^\circ$  (purple: mountain fold, orange: valley fold). (g) Two creased sheets combined by merging the nodes defining  $l_2$  and  $l'_2$ , along with a third node in each module marked in bright red. (h) A combined network of 12 sheets that sequentially collapses from the  $l^\circ$  to the  $l^\bullet$  lattice. (i) A 3D-printed planar module with two fixed points  $l^\bullet, l^\circ$ . Each module is composed of triangles connected by a thin layer of material, that (j) form a chain where (k) fixing the cyan hinge and pulling the red hinge yields a sequential transition from  $l^\bullet$  to  $l^\circ$ . (l) Photos of the quadrifolium and (m) chaotic networks made from cardstock bars held together by metal pins.

### 8.11. Constructing Physical Networks

Here, we implement this theory for designing the geometry of both the sequence and macroscopic structure of mechanical networks by constructing physical networks. We construct a super-stable and sequentially collapsible network by laser cutting the edges from 1/8-inch

thick acrylic, and connecting their joints using Chicago screws (Fig. 65a,b). Additionally, many deployable applications (Puig et al., 2010) require a compact initial geometry and a precise, rigid final geometry. Using wooden sticks that are joined by a staple prong at the joints, we show a 4-bar linkage with two fixed points  $l^\bullet$  and  $l^\circ$ , where the  $l^\bullet$  point is super-stable (Fig. 65d). These modules combine in a chain (Fig. 65e) that yields a wide spiral with an open channel at  $l^\bullet$ , and collapses to a narrow spiral with no channel at  $l^\circ$  (Fig. 65f, see the Appendix in Chapter 9 for the maps  $l_{k+1} = f(l_k)$ ).

To demonstrate the generalizability of our framework to 3-dimensional space, we model a creased square of paper as a linkage, where each crease is a rigid edge, and the intersection of creases is a node (Fig. 65f). We define  $l_1$  and  $l_2$  to be the distances between opposing corners in this sheet that collapses from the unfolded  $l^\bullet$  to the folded  $l^\circ$  crystalline states. We combine these modules by merging the nodes defining  $l_2$  and  $l'_2$  (Fig. 65g) to obtain an origami structure that collapses sequentially to a flat geometry.

These principles also extend to planar networks comprised of polygons (e.g., triangles) connected at vertices through a thin layer of flexible material (Fig. 65g). We designed a module with two fixed points  $l^\bullet$  and  $l^\circ$ , where the initial point  $l^\bullet$  is super-stable. We can chain these modules as before to yield the same iterated map  $l_{k+1} = f(l_k)$  (Fig. 65h), such that we obtain a sequential transition from  $l^\bullet$  to  $l^\circ$  by pulling on the network (Fig. 65i). Importantly, because this network is printed as shown, there is no required assembly (see the Appendix in Chapter 9 for the bond lengths of network and for considerations of elastic bond lengths).

## 8.12. Elasticity & Signal Propagation in the Mechanical AND gate

Until now, we have assumed that the bonds were rigid, or that any elastic deformations would decay to an equilibrium configuration of zero extension. However, real systems can sustain deformations from resistance to motion such as friction, which may alter the designed behavior of these networks. Here we provide two experiments to quantify the effect of bond

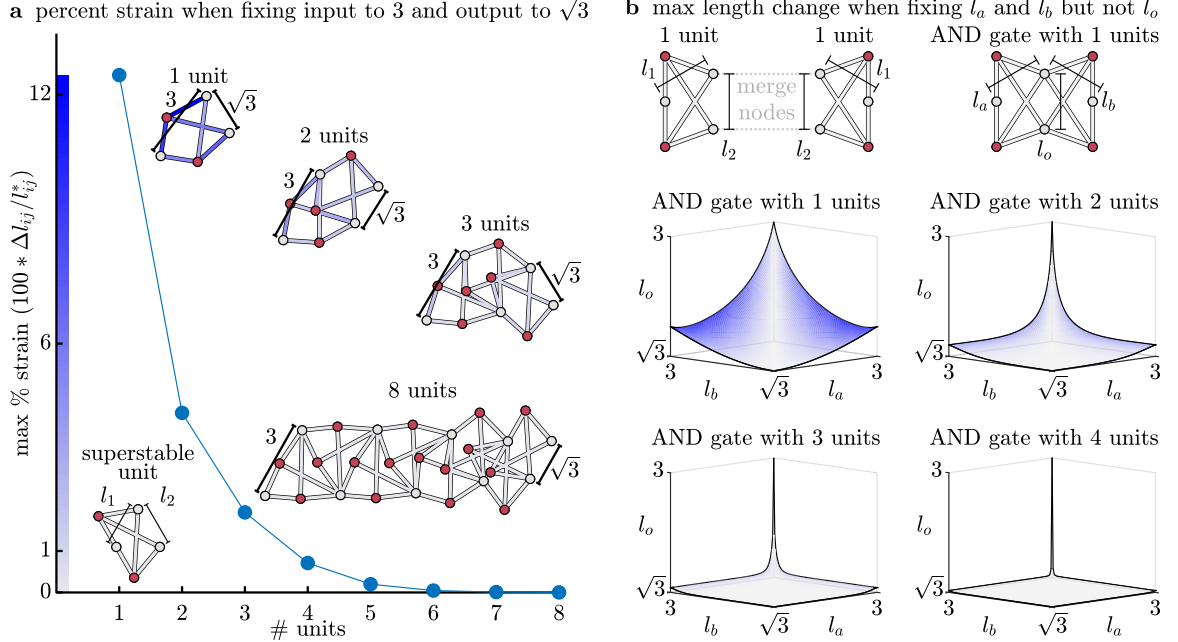


Figure 66: **Elastic Deformations in Superstable Networks.** (a) A plot of the maximum percent strain of the elastic bonds in networks comprising  $k$  superstable units. The initial length  $l_1 = 3$  is fixed to be open, the final length  $l_{k+1} = \sqrt{3}$  is fixed to be closed, and the minimum energy configuration is computed for  $k = \{1, 2, \dots, 8\}$ . (b) The nodes defining length  $l_{k+1}$  between two elastic network chains are merged to form an AND gate with input lengths  $l_a$  and  $l_b$ , and output length  $l_o$ . For  $k = \{1, 2, 3, 4\}$ , the input lengths are fixed at various values between  $\sqrt{3} \leq l_a, l_b \leq 3$ , and are plotted against the output length  $l_o$  and the maximum percent strain at the minimum energy configuration.

deformations.

the first experiment, we quantify the propagation of the input signal through a chain of superstable units with elastic bonds. We take the superstable unit shown in the bottom-left of Figure 66a, and replace the rigid bonds with elastic bonds of unit stiffness. Then, we force the input  $l_1$  open to 3, force the output  $l_2$  to  $\sqrt{3}$ , and compute the maximum bond strain at the equilibrium configuration. We repeat this procedure for longer chains comprising a range from 2 to 8 units, and plot the max strain (Fig. 66a). There is an appreciable strain up to around 5-6 units. Hence, enough of the input signal propagates to the output to appreciably deform the bonds. If the output were *not* fixed at  $\sqrt{3}$ , then the bonds must deform appreciably for the input signal to fully decay.

Importantly, this first result holds true for *any* scalar multiple of bond stiffness. Specifically, the bonds are modeled as linear springs with potential energy given by

$$V = \frac{1}{2} \sum_{(i,j) \in \mathcal{E}} k_{ij} (l_{ij}^* - l_{ij})^2, \quad (8.22)$$

where  $\mathcal{E}$  is the set of all bonds,  $k_{ij}$  is the stiffness of the bond connecting node  $i$  to node  $j$ ,  $l_{ij}^*$  is the equilibrium length of said bond, and  $l_{ij}$  is the current length of said bond. Because potential energy is linear in bond stiffness, the location of the minimum potential configuration and the percent bond strain do not change with scalar multiples of  $k_{ij}$ .

In the second experiment, we construct AND gates by combining chains of elastic super-stable units (Fig. 66b, top), and measure both the energetics and geometry of these AND gates. To directly quantify these effects, we construct AND gates by combining chains of 1 unit, 2 units, 3 units, and 4 units (Fig. 66b). We systematically fix the distance between the input nodes,  $l_a$  and  $l_b$ , across a range from  $\sqrt{3}$  to 3, and evolve the node coordinates to minimize the potential energy with all  $k_{ij} = 1$ . Then, we plot the distance between the output nodes,  $l_o$ , along with the maximum percent strain (Fig. 66b).

We find that an AND gate comprising 1 units has the greatest signal propagation with a large percent strain, but also has non-ideal geometric behavior. Specifically, the output fails to remain fully closed when only one of the two inputs are open (Fig. 66b). However, as we increase the number of units in a chain up to 4, we observe that the geometric changes much more closely resemble an ideal AND gate, where the output only opens appreciably when both  $l_a = l_b = 3$  are open.

### 8.13. Discussion

Ever-arising mechanical challenges (Sofla et al., 2010; Puig et al., 2010) drive the development of innovative designs (Overvelde et al., 2017; Wei et al., 2014; Cheung and Gershenfeld, 2013; Pellegrino, 2001), which in turn spark novel applications (Yang et al., 2015; Cummer

et al., 2016). In this work, we presented a simple theory for the principled design of a rich and complex set of folding sequences and large-scale geometries through the properties of a single module. Due to the practical and ubiquitous nature of linkages, these ideas are well-positioned to provide simple solutions to complex problems in robotic grasping (Yu Zheng and Wen-Han Qian, 2005), deployable mechanisms (Puig et al., 2010), morphing mechanical structures (Sofla et al., 2010), and tunable metamaterials (Liu and Semperlotti, 2018). By writing the large, non-linear geometric conformation of a network as the iteration of one module, we retain the richness of network motion while dramatically reducing design complexity.

Here, we studied the fundamental behaviors of this richness that directly arise from iterated maps. Immediate extensions include designing modules with complex maps (more than 2 fixed points, negative slopes at fixed points, critical slowing, bifurcations (Strogatz, 2018)), and developing principles for combining modules with different maps. The theory can also extend beyond iterated maps, where linkages follow a circular path that is not formally a function ( $d_2$  is not uniquely determined by  $d_1$ ). For ease of manufacturing, previous work on planar networks (Coulais et al., 2017) motivates the development of a module design framework specific to these systems. Finally, given the design framework for bistable linkages with elastic bonds (Kim et al., 2019b), a promising future direction lies in the design of sensors, adaptive response, and superelasticity seen in shape memory systems (Lendlein et al., 2005; Wei et al., 1998), deployable structures as seen in origami metamaterials and antenna (Silverberg et al., 2014b; Puig et al., 2010), and mechanical computation (Chen et al., 2021). Hence, this simple theory provides a versatile and unifying framework for designing large sequential conformational changes in mechanical networks.

CHAPTER 9 : Appendix to Nonlinear Dynamics & Chaos in Conformational  
Changes of Mechanical Metamaterials

### 9.1. Single Module Design: Infinitesimal Motion

In the main text, we design infinitesimal motions. Here we outline the design procedure.

We consider three nodes  $i \in \{1, 2, 3\}$  whose start positions,  $\mathbf{x}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$ , and infinitesimal

motions,  $d\mathbf{x}_i = \begin{bmatrix} dx_i \\ dy_i \end{bmatrix}$ , are fixed as constants. We consider additional nodes whose start positions  $\mathbf{x}_j$  and infinitesimal motions  $d\mathbf{x}_j$  are variables. The only edges are located between nodes  $i \in \{1, 2, 3\}$  and nodes  $j \in \{4, 5\}$  such that  $\mathcal{E} = \{1, 2, 3\} \times \{4, 5\}$ .

For either variable node  $j$ , we write the linearized constraints as

$$\mathbf{0} = \begin{bmatrix} (\mathbf{x}_1 - \mathbf{x}_j)^\top (d\mathbf{x}_1 - d\mathbf{x}_j) \\ (\mathbf{x}_2 - \mathbf{x}_j)^\top (d\mathbf{x}_2 - d\mathbf{x}_j) \\ (\mathbf{x}_3 - \mathbf{x}_j)^\top (d\mathbf{x}_3 - d\mathbf{x}_j) \end{bmatrix}.$$

Because the fixed nodes' positions and motions are constant, we expand each equation to pull out the variable node  $j$ 's position and motion

$$\mathbf{0} = \begin{bmatrix} \mathbf{x}_1^\top d\mathbf{x}_1 \\ \mathbf{x}_2^\top d\mathbf{x}_2 \\ \mathbf{x}_3^\top d\mathbf{x}_3 \end{bmatrix} - \begin{bmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \mathbf{x}_3^\top \end{bmatrix} d\mathbf{x}_j - \begin{bmatrix} d\mathbf{x}_1^\top \\ d\mathbf{x}_2^\top \\ d\mathbf{x}_3^\top \end{bmatrix} \mathbf{x}_j + \mathbf{x}_j^\top d\mathbf{x}_j \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix},$$

and notice that there is only 1 nonlinear term in this system, namely  $c = \mathbf{x}_j^\top d\mathbf{x}_j$ . If we temporarily omit this nonlinearity, and substitute  $c$  as a free variable, we can write the



linearized constraint equations as a linear system of equations  $\mathbf{b} = A\mathbf{v}$

$$\underbrace{\begin{bmatrix} \mathbf{x}_1^\top d\mathbf{x}_1 \\ \mathbf{x}_2^\top d\mathbf{x}_2 \\ \mathbf{x}_3^\top d\mathbf{x}_3 \end{bmatrix}}_{\mathbf{b}} = \underbrace{\begin{bmatrix} d\mathbf{x}_1^\top & \mathbf{x}_1^\top & -1 \\ d\mathbf{x}_2^\top & \mathbf{x}_2^\top & -1 \\ d\mathbf{x}_3^\top & \mathbf{x}_3^\top & -1 \end{bmatrix}}_A \underbrace{\begin{bmatrix} \mathbf{x}_j \\ d\mathbf{x}_j \\ c \end{bmatrix}}_{\mathbf{v}}. \quad (9.1)$$

Then, the variable node positions and motions arise as the particular solution  $\mathbf{v}_P = A^+\mathbf{b}$  (where  $A^+$  is the pseudo-inverse) and homogeneous solution  $\mathbf{v}_H \in \mathcal{N}(A)$  to Eq. 9.1 where

$$\mathbf{v} = \mathbf{v}_P + \mathbf{v}_H,$$

that satisfy the one nonlinear constraint  $c = \mathbf{x}_j^\top d\mathbf{x}_j$ . If a variable node that is connected to all fixed nodes is placed along this solution space, then the fixed nodes' positions  $\mathbf{x}_i$  and motions  $d\mathbf{x}_i$  satisfy the edge constraints.

## 9.2. Single Module Design: Finite Displacement

In the main text, we also design finite motions. Here we outline the design framework. We

consider three nodes  $i \in \{1, 2, 3\}$ , whose start positions,  $\mathbf{x}_i^\bullet = \begin{bmatrix} x_i^\bullet \\ y_i^\bullet \end{bmatrix}$ , and end positions,

$\mathbf{x}_i^\circ = \begin{bmatrix} x_i^\circ \\ y_i^\circ \end{bmatrix}$ , are fixed as constants. We consider additional nodes  $j \in \{4, 5\}$  whose start positions  $\mathbf{x}_j^\bullet$  and end positions  $\mathbf{x}_j^\circ$  are variables. The only edges are located between nodes  $i$  and nodes  $j$ , such that the edges of the entire unit are given by  $\mathcal{E} = \{1, 2, 3\} \times \{4, 5\}$ . For both initial and final positions to be reachable by the same unit with rigid edges, we must ensure that the edge lengths are equal at both positions. Hence, for either of the nodes

$j \in \{4, 5\}$ , we satisfy the constraint that the edge lengths at both positions are equal:

$$\begin{bmatrix} (\mathbf{x}_1^\bullet - \mathbf{x}_j^\bullet)^\top (\mathbf{x}_1^\bullet - \mathbf{x}_j^\bullet) \\ (\mathbf{x}_2^\bullet - \mathbf{x}_j^\bullet)^\top (\mathbf{x}_2^\bullet - \mathbf{x}_j^\bullet) \\ (\mathbf{x}_3^\bullet - \mathbf{x}_j^\bullet)^\top (\mathbf{x}_3^\bullet - \mathbf{x}_j^\bullet) \end{bmatrix} = \begin{bmatrix} (\mathbf{x}_1^\circ - \mathbf{x}_j^\circ)^\top (\mathbf{x}_1^\circ - \mathbf{x}_j^\circ) \\ (\mathbf{x}_2^\circ - \mathbf{x}_j^\circ)^\top (\mathbf{x}_2^\circ - \mathbf{x}_j^\circ) \\ (\mathbf{x}_3^\circ - \mathbf{x}_j^\circ)^\top (\mathbf{x}_3^\circ - \mathbf{x}_j^\circ) \end{bmatrix}.$$

We expand these terms to yield

$$\begin{bmatrix} \mathbf{x}_1^{\bullet\top} \mathbf{x}_1^\bullet \\ \mathbf{x}_2^{\bullet\top} \mathbf{x}_2^\bullet \\ \mathbf{x}_3^{\bullet\top} \mathbf{x}_3^\bullet \end{bmatrix} - 2 \begin{bmatrix} \mathbf{x}_1^{\bullet\top} \\ \mathbf{x}_2^{\bullet\top} \\ \mathbf{x}_3^{\bullet\top} \end{bmatrix} \mathbf{x}_j^\bullet + \mathbf{x}_j^{\bullet\top} \mathbf{x}_j^\bullet \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1^{\circ\top} \mathbf{x}_1^\circ \\ \mathbf{x}_2^{\circ\top} \mathbf{x}_2^\circ \\ \mathbf{x}_3^{\circ\top} \mathbf{x}_3^\circ \end{bmatrix} - 2 \begin{bmatrix} \mathbf{x}_1^{\circ\top} \\ \mathbf{x}_2^{\circ\top} \\ \mathbf{x}_3^{\circ\top} \end{bmatrix} \mathbf{x}_j^\circ + \mathbf{x}_j^{\circ\top} \mathbf{x}_j^\circ \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix},$$

and rearrange to isolate the constants  $\mathbf{x}_i^\bullet, \mathbf{x}_i^\circ$  from the variables  $\mathbf{x}_j^\bullet, \mathbf{x}_j^\circ$  to yield another linear equation, with a similarly substituted nonlinear free variable  $c = \mathbf{x}_j^{\bullet\top} \mathbf{x}_j^\bullet - \mathbf{x}_j^{\circ\top} \mathbf{x}_j^\circ$

$$\underbrace{\begin{bmatrix} \mathbf{x}_1^{\bullet\top} \mathbf{x}_1^\bullet - \mathbf{x}_1^{\circ\top} \mathbf{x}_1^\circ \\ \mathbf{x}_2^{\bullet\top} \mathbf{x}_2^\bullet - \mathbf{x}_2^{\circ\top} \mathbf{x}_2^\circ \\ \mathbf{x}_3^{\bullet\top} \mathbf{x}_3^\bullet - \mathbf{x}_3^{\circ\top} \mathbf{x}_3^\circ \end{bmatrix}}_{\mathbf{b}} = \underbrace{\begin{bmatrix} 2\mathbf{x}_1^{\bullet\top} & -2\mathbf{x}_1^\circ & -1 \\ 2\mathbf{x}_2^{\bullet\top} & -2\mathbf{x}_2^\circ & -1 \\ 2\mathbf{x}_3^{\bullet\top} & -2\mathbf{x}_3^\circ & -1 \end{bmatrix}}_A \underbrace{\begin{bmatrix} \mathbf{x}_j^\bullet \\ \mathbf{x}_j^\circ \\ c \end{bmatrix}}_{\mathbf{v}}. \quad (9.2)$$

Again, we see that  $\mathbf{b}$  and  $A$  only contain fixed node initial and final positions that are constants, such that the equation is linear in our variable node start and end positions,  $\mathbf{v}$ . Our solution space is again given by a particular and homogeneous solution  $\mathbf{v} = \mathbf{v}_P + \mathbf{v}_H$  that satisfies one quadratic constraint  $c = \mathbf{x}_j^{\bullet\top} \mathbf{x}_j^\bullet - \mathbf{x}_j^{\circ\top} \mathbf{x}_j^\circ$ . By placing unspecified nodes and edges on this space such that  $M = 4$ , we have 1 conformational motion where the desired start  $\mathbf{x}^\bullet$  and end  $\mathbf{x}^\circ$  positions have the same edge lengths. Because the additional constraint  $c$  is a quadratic constraint, our solution space takes the form of a conic section.

### 9.3. Analytic Form of the Iterated Map

In the main text, we use the analytical form of the iterated map. Whereas numerical integration yields points along the 1-dimensional map  $f$  of conformational motions over

time given by  $l_1(t)$  and  $l_2(t)$ , an analytical form of the map is desired for numerically sensitive applications such as computing the Lyapunov exponent of a map. Specifically, we would like the equational form of our map  $f$  between the lengths across unconnected nodes for  $l_2 = f(l_1)$ . We solve for this equational form in our system of 5 nodes (3 fixed, 2 variable) and 6 edges. Without loss of generality, we number the fixed nodes such that  $l_1$  is the distance between nodes 1 and 2, and  $l_2$  is the distance between nodes 2 and 3, and we number the variable nodes as 4 and 5. Further, we can always find a set of rigid body motions to place node 2 at the coordinates  $(x_2, y_2) = (0, 0)$ , and place node 1 at the coordinates  $(x_1, y_1) = (l_1, 0)$ . We denote the length of an edge between nodes  $i$  and  $j$  as  $l_{ij}$ .

We begin by writing the variable node positions as a function of  $(x_1, y_1)$  and  $(x_2, y_2)$  through the distance constraints from the edge lengths. Specifically, nodes  $j = 4, 5$  satisfy

$$\begin{aligned} l_{1j}^2 &= (x_j - x_1)^2 + (y_j - y_1)^2 = x_j^2 + y_j^2 - 2l_1x_j + l_1^2, \\ l_{2j}^2 &= (x_j - x_2)^2 + (y_j - y_2)^2 = x_j^2 + y_j^2, \end{aligned}$$

and subtracting the two equations, we can solve for  $x_j$ , and resubstitute to solve for  $y_j$

$$x_j = \frac{l_1^2 + l_{2j}^2 - l_{1j}^2}{2l_1} \quad y_j = \pm \sqrt{l_{2j}^2 - \left( \frac{l_1^2 + l_{2j}^2 - l_{1j}^2}{2l_1} \right)^2}.$$

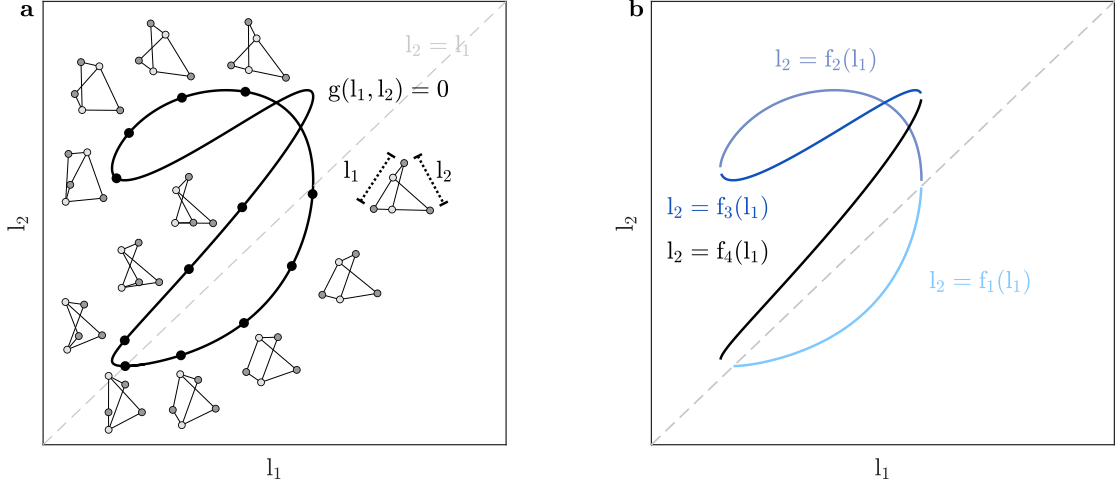
From the known initial position of the nodes,  $\mathbf{x}^0$ , we can determine whether  $y_j$  is positive or negative. Then, we can write the position of node 3 as satisfying constraints

$$\begin{aligned} l_{34}^2 &= (x_3 - x_4)^2 + (y_3 - y_4)^2 \\ l_{35}^2 &= (x_3 - x_5)^2 + (y_3 - y_5)^2, \end{aligned}$$

which are the equations for two circles: one centered at  $(x_4, y_4)$  with radius  $l_{34}$  and another centered at  $(x_5, y_5)$  with radius  $l_{35}$ . The intersection of these circles yields  $x_3$  and  $y_3$ , and is solved symbolically as a function of  $l_1$ . The initial position of node 3 is used to determine

which of the generally two points of intersection are used. Because  $l_2$  is the distance between nodes 2 and 3, and node 2 is located at  $(0,0)$ , the distance  $l_2$  is simply the length of the position of node 3, finally yielding the map  $l_2 = f(l_1) = \sqrt{x_3^2(l_1) + y_3^2(l_1)}$ .

#### 9.4. Condition for a Conformational Motion to Act as a Map



**Figure 67: Decomposition of a Conformational Motion Into Valid Map Segments.** (a) A plot of distances  $l_1$  versus  $l_2$  between unconnected red nodes in a module along the full conformational motion, with example network geometries at select points. (b) Decomposition of the curve  $(l_1, l_2)$  into four segments that uniquely map distance  $l_1$  to distance  $l_2$ .

Throughout the main text, we consider the implications of viewing module combinations as iterated maps. For this perspective to hold true, we must write  $l_2$  as a proper function of  $l_1$ , where each value of  $l_1$  uniquely determines a value of  $l_2$ . This property generally does not hold true along the full trajectory of a single module. For example, consider the following module (Fig. 67a), with 1 conformational motion, and distances  $l_1$  and  $l_2$  between unconnected red nodes. As the node coordinates change along this conformational motion, we can plot the distance pair  $(l_1, l_2)$  as a continuous 1-dimensional curve that is some function of  $l_1$  and  $l_2$ , shown in black. Immediately, we notice that for many values of  $l_1$ , there are multiple possible values of  $l_2$  on the curve. For the iterated map perspective to hold true, we must select a continuous segment of this curve where  $l_2$  is uniquely determined

by  $l_1$ . For this particular module, we can select 4 such segments  $f_1$  (light blue),  $f_2$  (blue),  $f_3$  (dark blue),  $f_4$  (black) as 4 valid maps from  $l_1$  to  $l_2$  (Fig. 67b). We ensure that the combined networks of the main text are valid functions.

### 9.5. Numerically Characterizing Chaos: Lyapunov Exponent

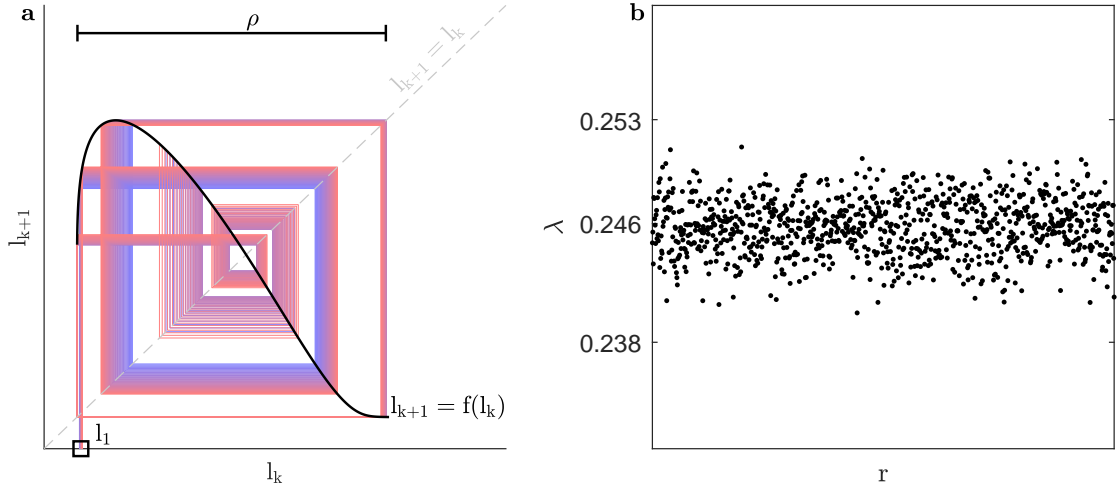


Figure 68: **Trajectory Divergence at Chaos.** (a) Map  $l_{k+1} = f(l_k)$  of the chaotic module, starting at 50 evenly spaced lengths from  $0.4224 \lesssim l_1 \lesssim 0.4324$ , whose trajectories diverge. The map's domain and range are bounded by a range of lengths  $\rho$ . (b) We sample 1000 evenly spaced points along  $r$  at  $0.4155 \lesssim l_1 \lesssim 1.4842$  as initial lengths. Then we iterate our map 10000 times to settle the lengths onto the attractor, and compute the Lyapunov exponent for a subsequent 50000 iterations.

In the main text, we show a chaotic network whose map between distances  $l_2 = f(l_1)$  is a general tent map demonstrating chaotic behavior. We quantify this behavior using the Lyapunov exponent that measures the sensitivity of the trajectory to minute changes in initial conditions. We iterate the map  $n$  times for an initial distance  $l_1$  and a small perturbation  $l'_1 = l_1 + \delta_1$ , such that  $l_{n+1} = f^n(l_1)$ , and  $l'_{n+1} = f^n(l'_1)$ . Then, we measure how far the trajectories have diverged as  $\delta_{n+1} = l'_{n+1} - l_{n+1}$ . The Lyapunov exponent captures the exponential rate,  $\lambda$ , at which this divergence occurs according to  $|\delta_{n+1}| = |\delta_1|e^{n\lambda}$ . For the trajectory of distances  $l_1, l_2, \dots$  at the limit of  $n \rightarrow \infty$ , this exponent is defined as the

average log of the slope at each distance

$$\lambda = \lim_{n \rightarrow \infty} \left( \frac{1}{n} \sum_{i=1}^n \ln |f'(l_i)| \right).$$

Across 1000 evenly spaced  $l_1$  across the full map range, we find a tight distribution of positive exponents averaged around  $\lambda \approx 0.246 > 0$ , which is a hallmark of chaotic systems.

## 9.6. Maps of Physical Linkage & 3D-Printed Modules

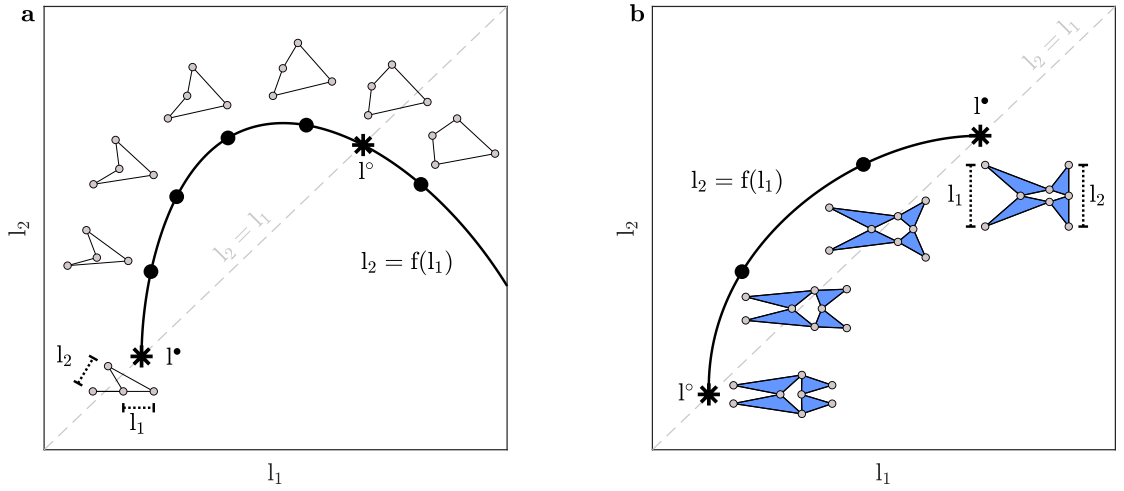


Figure 69: **Maps of Physical Networks.** (a) Map  $l_2 = f(l_1)$  of the 4-bar linkage used as a deployable example, with a super-unstable fixed-point  $l^\bullet$  and a stable fixed-point  $l^\circ$ . (b) Map  $l_2 = f(l_1)$  of the planar network, with a super-stable fixed-point  $l^\bullet$  and a super-unstable fixed-point  $l^\circ$ .

In the main text, we construct physical networks using our linkage design framework. In addition, we show a 4-bar linkage and a 3D printed planar network. Here we show the maps  $l_2 = f(l_1)$  for these additional networks. For the 4-bar linkage, we begin in an initial fixed-point collapsed state  $l^\bullet$  that is super-unstable (slope =  $\infty$ ), where an infinitesimal change in  $l_2$  produces no change in  $l_1$ . Further along the motion, we get a stable fixed point  $l^\circ$  (Fig. 69a). As for the planar network, we can model each triangular face as a triangular linkage, and plot the map of  $l_2 = f(l_1)$  along the conformational motion. This module is super-stable at the initial fixed point  $l^\bullet$ , and is additionally super-unstable at the final fixed

point  $l^\circ$ .

### 9.7. Construction & Map of Origami Module

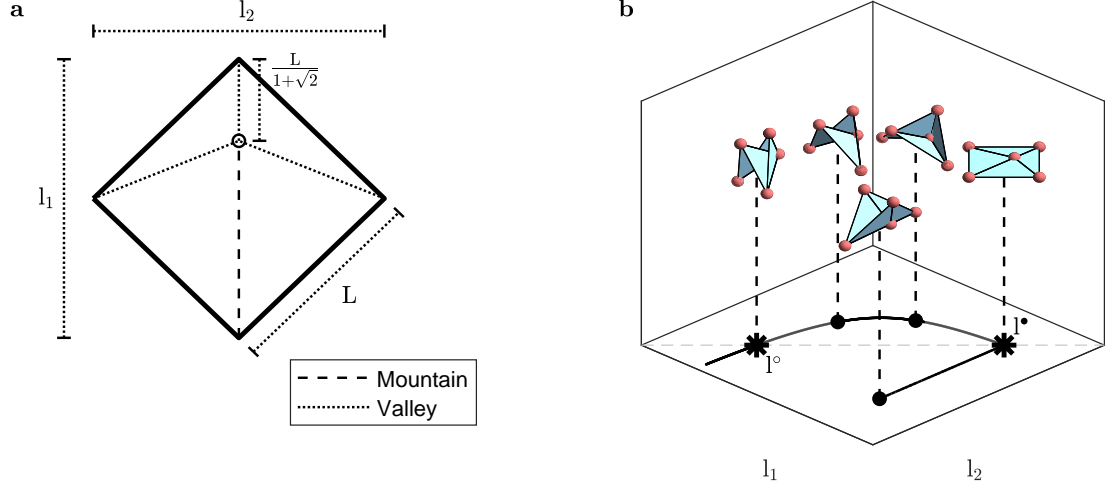


Figure 70: **Origami Sheet Construction & Map.** (a) Construction of creases in square origami sheet, with labeled dimensions and with mountain and valley folds. We label  $l_1$  and  $l_2$  as the distance between the corresponding corners. (b) Map  $l_2 = f(l_1)$  of this origami sheet that has two fixed points:  $l^\bullet$  corresponds to the flat sheet configuration, and  $l^\circ$  corresponds to a folded configuration. The flat sheet configuration sits at a kinematic bifurcation, where another trajectory exists in which the network simply folds along the main diagonal crease.

In the main text, we demonstrate the use of our framework in an origami unit. Here we detail the construction of the unit shown in the main text, and numerically plot its map. We begin with a square of paper with side length  $L$ , and crease it once along the diagonal. We label  $l_1$  and  $l_2$  as the distances between the opposite marked corners of this square, and also label the mountain and valley creases (Fig. 70 a). We then show the map of  $l_1$  *versus*  $l_2$  along the 1 conformational motion, with the corresponding network geometries shown above sampled points along the map (Fig. 70b). At the flat sheet configuration, the network exists at a kinematic bifurcation, such that other possible trajectories exist, such as the network folding along the main diagonal crease with both mountain or valley folds.

## 9.8. Edge Lengths are Determined by Added Node Placement

Throughout the main text, our unit design process involves *first* placing the added nodes 4 and 5 on the solution space, and *second* connecting the added nodes to nodes 1, 2, and 3 at their start positions. This ordering means that the lengths of the rigid edges are determined by our placement of the added nodes 4 and 5 on the solution space. In every designed unit, there are only 6 edges, between node pairs: (1, 4), (2, 4), (3, 4), (1, 5), (2, 5), and (3, 5). As an example, consider the unit design of three nodes, 1, 2, and 3, where we desire for their end positions to move radially outward from their start positions (Fig. 71a).

In one scenario, we first choose to add nodes 4 and 5 somewhere on the solution space, and then connect them to nodes 1, 2, and 3 with edges of the appropriate length (Fig. 71b). In another scenario, we first choose to add nodes 4 and 5 at a *different* location on the solution space, and then connect them to nodes 1, 2, and 3 with edges of *different* appropriate lengths (Fig. 71c). We observe this phenomenon again in yet another choice of placements for nodes 4 and 5 on the solution space (Fig. 71d).

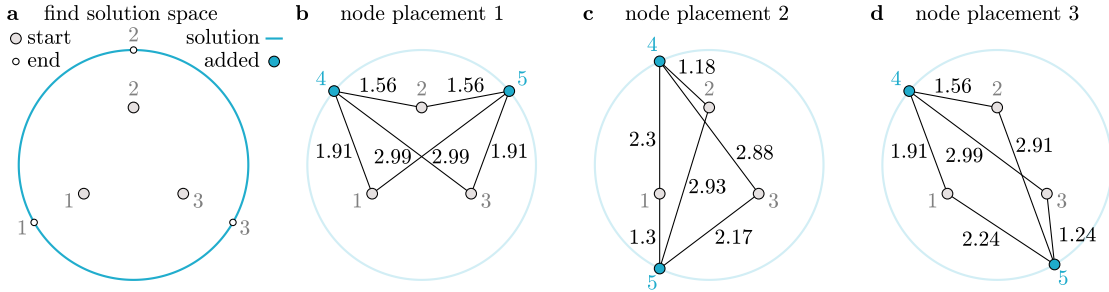


Figure 71: **Edge Lengths are Determined by Added Node Placement.** (a) Three nodes, 1, 2, and 3, whose end positions are to be designed to move radially outwards, with the solution space shown in blue. (b) The unit design process involves first adding nodes 4 and 5 (in blue) on the solution space, and then connecting them with edges of labeled lengths to nodes 1, 2, and 3. (c,d) Placing the added nodes at different locations on the solution space yields different edge lengths.

Hence, the lengths of the rigid edges are not chosen first. Rather, the locations of the added nodes, 4 and 5, are chosen first to be on the solution space. Then, they are connected to nodes 1, 2, and 3 with rigid edges of the appropriate length. Specifically, they are



connected to nodes 1, 2, and 3 by edges that have length equal to the distance between the start positions of nodes 1, 2, and 3, and the positions of added nodes 4 and 5.

### 9.9. Combining Units Only Merges Nodes Between Units

In the main text, we describe how we combine units by merging the nodes between the units. Here we will describe this procedure in more depth. Consider the two units shown in Figure 72a, unit  $k$  and unit  $k + 1$ , where each unit has three non-added nodes, 1, 2, and 3 (in gray), and two added nodes, 4 and 5 (in blue). As per the methods in the main text, for unit  $k$ , we define lengths  $l_k$  and  $l_{k+1}$  as the lengths between unconnected node pairs (1,2) and (2,3), respectively. Similarly for unit  $k + 1$ , we define lengths  $l_{k+1}$  and  $l_{k+2}$  as the lengths between unconnected node pairs (1,2) and (2,3), respectively. We first take the mirror image of unit  $k + 1$  by flipping it along the horizontal axis (Fig. 72b), as this is the orientation of units  $k$  and  $k + 1$  before they are combined throughout the main text.

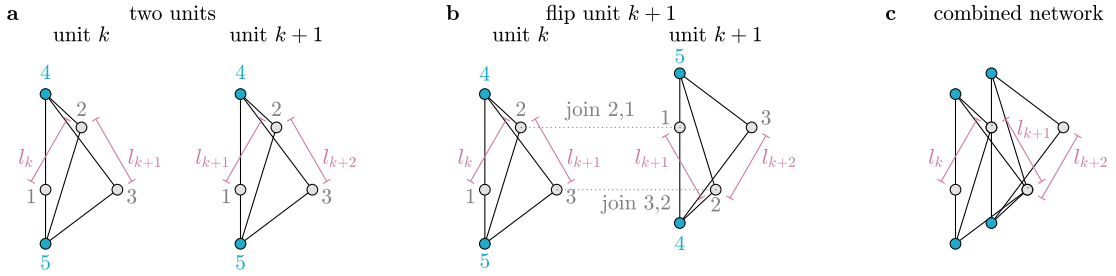


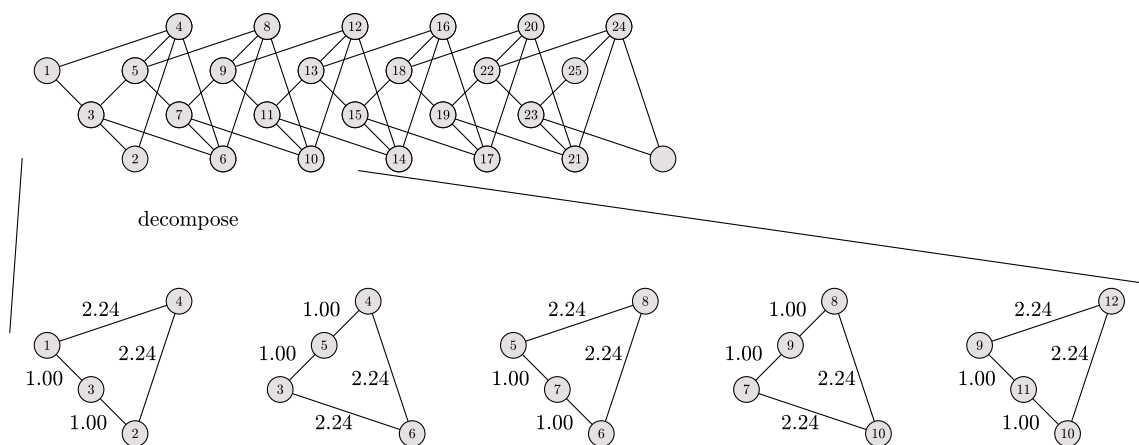
Figure 72: **Combining Units Only Merges Nodes Between Units.** (a) Two units,  $k$  and  $k + 1$ , that each have nodes 1 – 5. Unit  $k$  has lengths  $l_k$  and  $l_{k+1}$ , and unit  $k + 1$  has lengths  $l_{k+1}$  and  $l_{k+2}$ . (b) Unit  $k + 1$  is flipped along the horizontal axis to orient it according to our main text examples, and the nodes marked by the dotted lines are merged to form (c) the combined network.

We combine unit  $k$  to unit  $k + 1$  by merging nodes, and describe the process in three different ways. Visually, in Figure 72b, the nodes connected by the dotted lines are merged, which can be visualized as bringing the units closer together until the nodes connected by the dotted line overlap, and are then glued together to form Figure 72c. Equationally, we notice that length  $l_{k+1}$  is used to define two lengths: the length between node pair (2,3)

in unit  $k$ , and the length between node pair  $(1, 2)$  in unit  $k + 1$ . The nodes defining the common length  $l_{k+1}$  are brought together and merged. Verbally, we merge unit  $k$  to unit  $k + 1$  by merging node 2 of unit  $k$  to node 1 of unit  $k + 1$ , and by merging node 3 of unit  $k$  to node 2 of unit  $k + 1$ . No other modifications are made when combining units, and no edges are added.

## 9.10. Edge Lengths of All Network Examples

Throughout the main text, we perform many constructions from units to network chains. To drive additional intuition, we show a deconstruction from network chains to units, and write down the edge lengths of units. Importantly, here we will use a node numbering scheme that is different from that used in the main text. In the main text, we numbered the nodes of each *unit*, and merged the nodes together. Here, we will number the nodes of the *network*, and preserve this numbering throughout the deconstruction.



**Figure 73: Decomposition and Edge Lengths of the 4-Bar Linkage Example.** Top: the 4-bar linkage network from the main text with unique node labels. Bottom: decomposition of the network into the composite 4-bar linkage units, with labeled edge lengths and preserved node labels. For example, node 5 connects to nodes 3, 4, and 7 *via* edges of length 1.00, and node 8 *via* an edge of length 2.24.

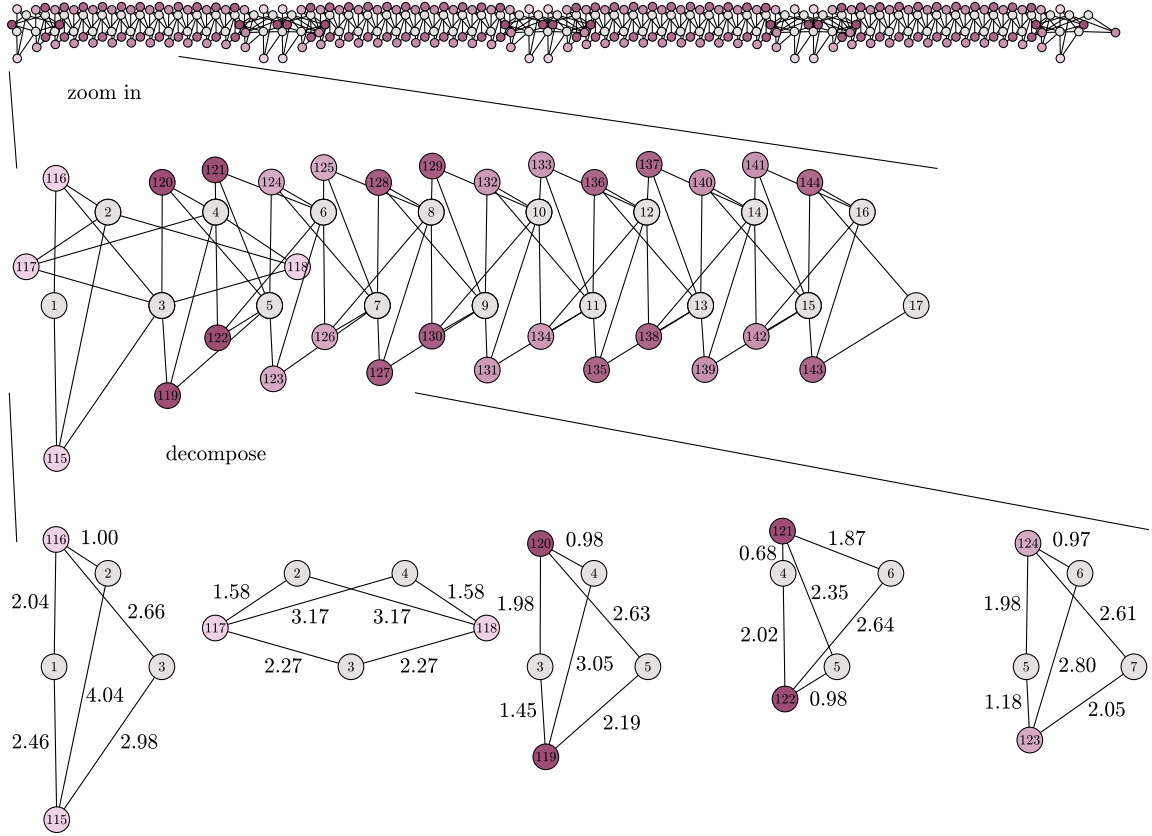


Figure 74: **Decomposition and Edge Lengths of the Designed Quadrifolium.** Top: the quadrifolium network from the main text. Middle: a zoomed-in view of a subset of the network with uniquely labeled nodes. Bottom: decomposition of the network into the composite designed units, with labeled edge lengths and preserved node labels. For example, node 4 connects to node 117 *via* an edge of length 3.17, to node 118 *via* an edge of length 1.58, to node 119 *via* an edge of length 3.05, to node 120 *via* an edge of length 0.98, to node 121 *via* an edge of length 0.68, and to node 122 *via* an edge of length 2.02.

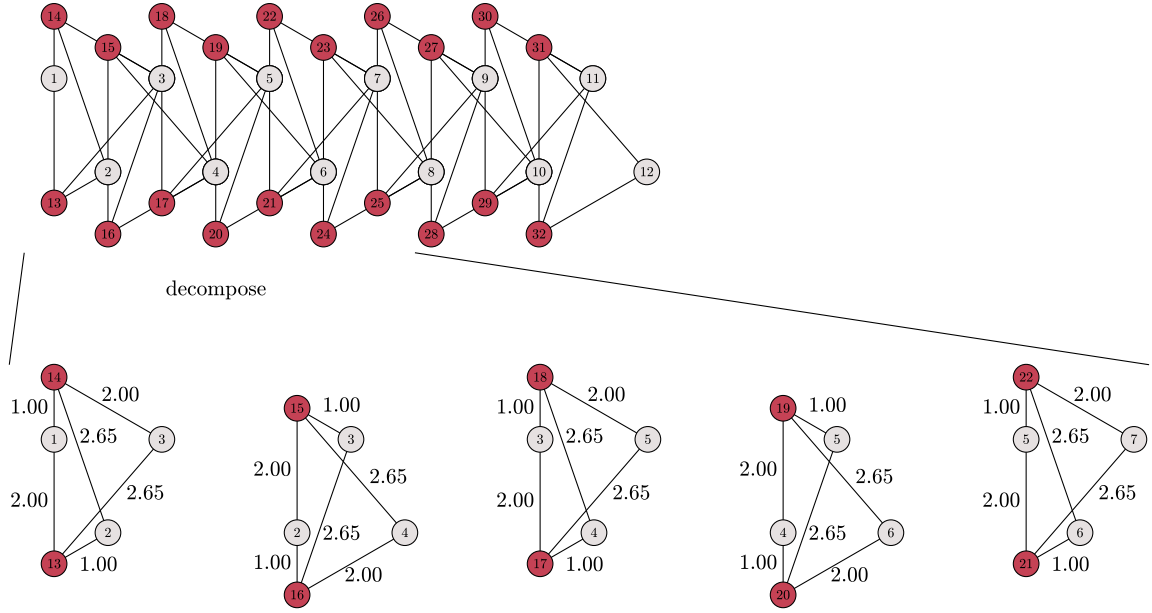
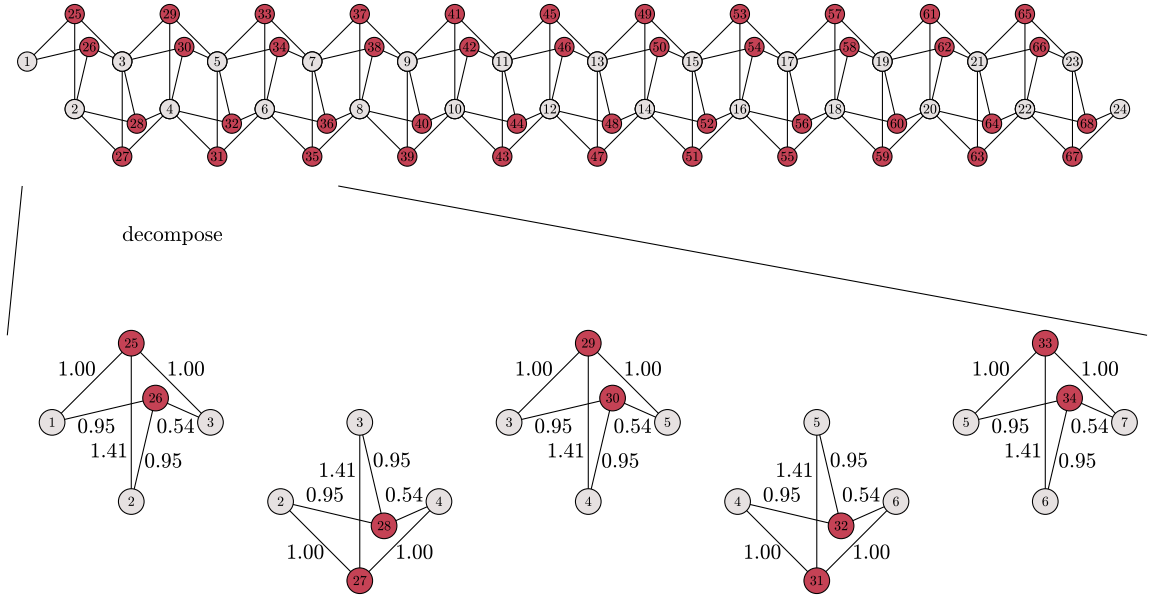


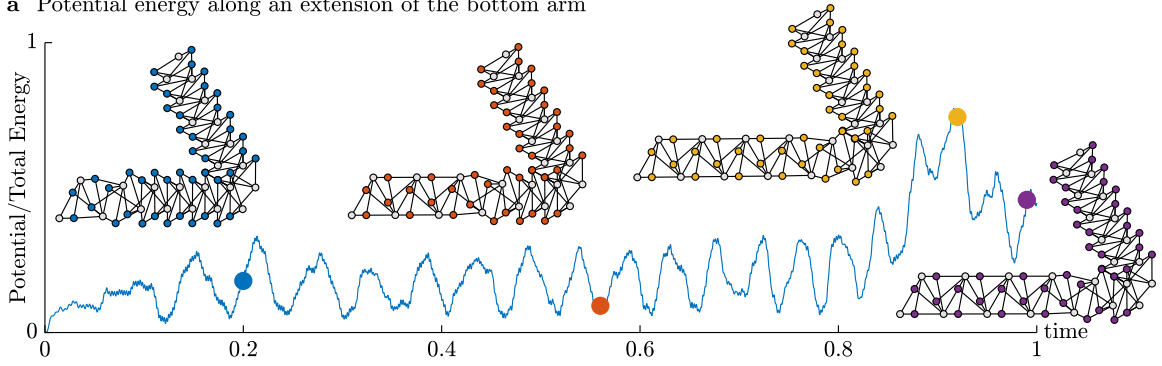
Figure 75: **Decomposition and Edge Lengths of the Designed Superstable Network.** Top: the superstable network from the main text with uniquely labeled nodes. Bottom: decomposition of the network into the composite designed units, with labeled edge lengths and preserved node labels. For example, node 4 connects to node 15 *via* an edge of length 2.65, to node 16 *via* an edge of length 2.00, to node 17 *via* an edge of length 1.00, to node 18 *via* an edge of length 2.65, to node 19 *via* an edge of length 2.00, and to node 20 *via* an edge of length 1.00.



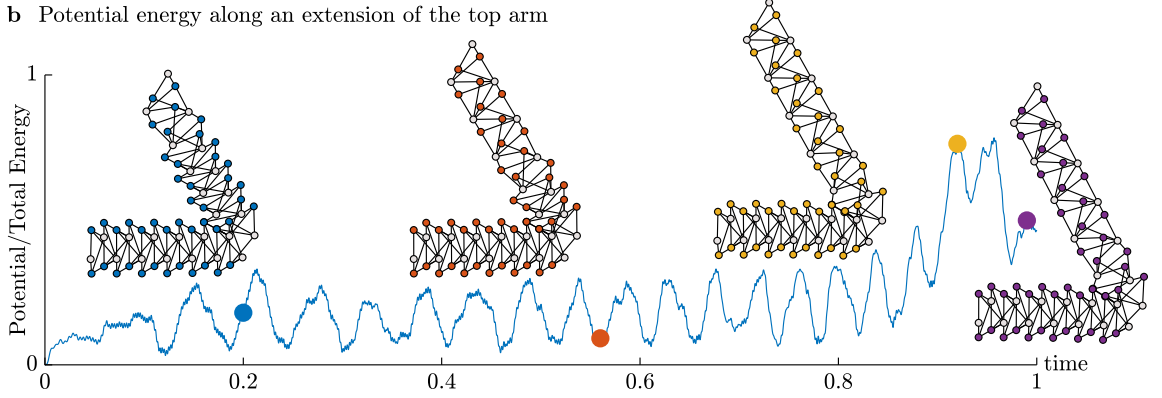
**Figure 76: Decomposition and Edge Lengths of the Designed Chaotic Network.** Top: the chaotic network from the main text with uniquely labeled nodes. Bottom: decomposition of the network into the composite designed units, with labeled edge lengths and preserved node labels. For example, node 4 connects to node 27 *via* an edge of length 1.00, to node 28 *via* an edge of length 0.54, to node 29 *via* an edge of length 1.41, to node 30 *via* an edge of length 0.95, to node 31 *via* an edge of length 1.00, and to node 32 *via* an edge of length 0.95.

### 9.11. Elasticity in the Superstable Mechanical AND Gate

**a** Potential energy along an extension of the bottom arm



**b** Potential energy along an extension of the top arm



**Figure 77: Evolution of an Elastic Mechanical AND Gate.** (a) Potential energy over time of a mechanical AND gate where the rigid rods are replaced with elastic rods, with an initial velocity of the left-most gray node going to the left. After the initial velocity  $t = 0$ , there is no more energy that is input into the system, and no other external forces that act on the network, such that the total energy remains constant over time. (b) The same simulation is run for the same network, except the initial velocity is now on the top-most gray node, moving to the top-left.

In the main text, we construct a mechanical AND gate using units at a superstable fixed point that converge to the fixed point quadratically. In our exposition, we often refer to very small quantities such as  $10^{-64}$ , which brings up the potential for the sensitivity of the AND gate to factors such as elasticity. To study the effect of elasticity, we examine the behavior of the mechanical AND gate after replacing the rigid bonds with elastic springs characterized by unit spring constant and node mass.

In the first initial condition, the velocity of the left-most gray node is set to move to the left, and the system's Hamiltonian is evolved over time with no external forces (Fig. 77a) As

can be seen, the bottom arm is allowed to extend independently of the top arm, but hits an energy barrier once the motion reaches the intersection of the arms (Fig. 77a, yellow). The motion is no longer allowed to propagate beyond this point. In the second initial condition, the velocity of the top-most gray node is set to move to the top-left, and the system's Hamiltonian is evolved over time again with no external forces (Fig. 77b). As before, the top arm is allowed to extend independently, but hits an energy barrier once the motion reaches the intersection (yellow), and is no longer allowed to propagate. Hence, the elastic network demonstrates the same qualitative behavior as the rigid network, whereby both arms are allowed to move independently, but the signal is unable to propagate until both arms have fully extended.

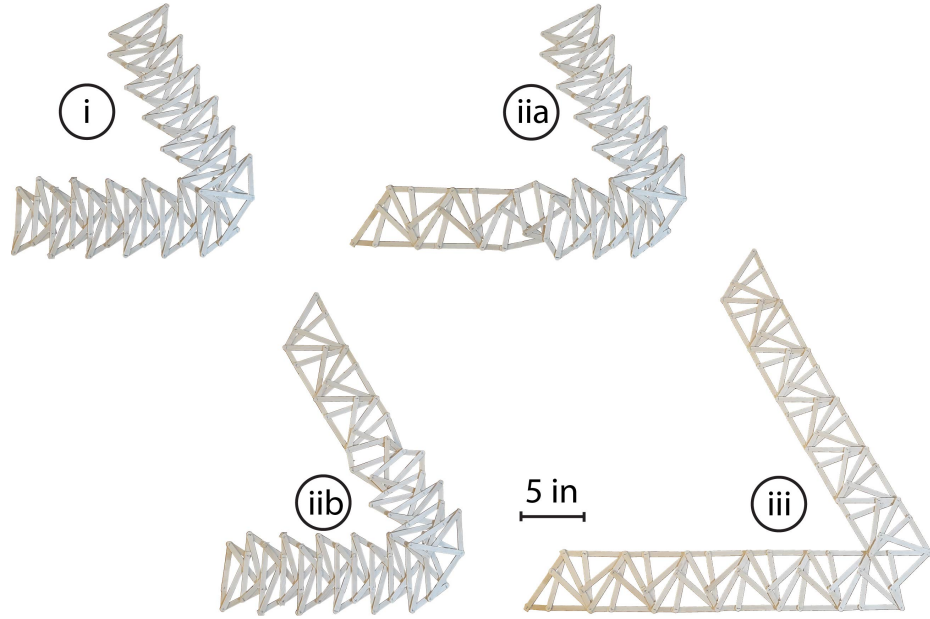


Figure 78: **Physical Construction of the Mechanical AND Gate.**

To further demonstrate the ability of the theory to be physically implemented, we physically construct the mechanical AND gate out of wooden sticks to reproduce Figure 11 of the main text.

### 9.12. Elaboration of Unit Cell Topology

Throughout the main text and supplement, we provide one perspective on the creation of unit cells. Specifically, we take an additive perspective whereby we begin with a designed 5-node unit, and combine multiple 5-node units together by merging node pairs. Here, we put forth another perspective that was very kindly written by a reviewer which considers a pre-constructed lattice of already-combined units.

Consider a particle in the two-dimensional plane attached to a particle at  $\mathbf{v}_1$  by a rod of length  $a$  and to a particle at  $\mathbf{v}_2$  by a rod of length  $b$ . This particle is thus constrained to lie at one of two points, which may be expressed as

$$\mathbf{v}_m^\pm = \mathbf{v}_1 + f_\parallel(a, b, |\mathbf{v}_2 - \mathbf{v}_1|)\hat{v}_\parallel \pm f_\perp(a, b, |\mathbf{v}_2 - \mathbf{v}_1|)\hat{v}_\perp. \quad (9.3)$$

Here,  $f_\parallel, f_\perp$  may be given explicitly in terms of simple algebraic functions.  $\hat{v}_\parallel$  is a unit vector pointing from the first to the second particle and  $\hat{v}_\perp$  is this vector rotated by a quarter turn in the counter-clockwise direction.

A unit cell applies this procedure three times, such that particles initially at positions  $\mathbf{r}_1^k, \mathbf{r}_2^k$  define the unit cell.  $\mathbf{r}_3^k$  is connected to the first two particles by rods of length  $a_1, b_1$ , while  $\mathbf{r}_4^k$  is connected to the first two particles by rods of length  $a_2, b_2$ . Finally,  $\mathbf{r}_5^k$  is connected to the third and fourth particles by rods of length  $a_3, b_3$ . The rod lengths are as given in the figure in the Supplemental section 9.10. In each cell, the mapping used is either  $\pm, =, +$ , or  $\pm, =, -$ , with the choice alternating between subsequent cells. The length associated with a cell is defined as  $l_k \equiv |\mathbf{r}_2^k - \mathbf{r}_1^k|$ . The second and the fifth particles in the cell form the first two in the next cell:  $\mathbf{r}_1^{k+1} = \mathbf{r}_2^k, \mathbf{r}_2^{k+1} = \mathbf{r}_5^k$ . Note that creating each new cell places three additional particles in the two-dimensional plane with six additional rod constraints, generating an isostatic structure that maps lengths via some complicated functional relationship  $l_{k+1} = f(l_k)$ .



### 9.13. Chaos Remains Generic to Small Changes in Bond Length

In the main text, we demonstrate several examples of chaotic networks through either a period-doubling route to chaos or through a unit displaying a 3-cycle. Here, we test the robustness of chaos to noise. Specifically, we begin with a network that is known to produce chaos (Fig. 79, left), and test the preservation of chaos under perturbations.

First, we take the network in Fig. 79, with the displayed node coordinates. Specifically, node 1 is positioned at  $(-0.93, -0.90)$ , node 2 is positioned at  $(0, 0)$ , node 3 is positioned at  $(0.93, -0.90)$ , node 4 is positioned at  $(0, -0.94)$ , and node 5 is positioned at  $(-0.62, -0.88)$ . The following pairs of nodes are connected by rigid bonds:  $\mathcal{E} = \{(1, 4), (2, 4), (3, 4), (1, 5), (2, 5), (3, 5)\}$ . The length of the bond between node  $i$  and node  $j$  is fixed to be the distance between node  $i$  and node  $j$ . The length  $l_k$  is defined as the distance between the unconnected node pair  $(1, 2)$ , and the length  $l_{k+1}$  is defined as the distance between the unconnected node pair  $(2, 3)$ . This network has one conformational motion, and plotting the distance  $l_{k+1}$  against the distance  $l_k$  along this conformational motion yields the blue curve (Fig. 79, left). This curve yields a map  $f$  that maps  $l_{k+1} = f(l_k)$ . The Lyapunov exponent of this map is computed by first choosing  $l_1 = 1.25$  as an initial condition, then iterating the map 10,000 times *without* keeping track of the slope of the map to remove any dependencies on the initial condition, and finally by computing the Lyapunov exponent on a subsequent 10,000 iterations. The Lyapunov exponent we obtained was approximately 0.4874, which is greater than 0, such that the map is chaotic.

To test whether chaos is a generic property, we first removed the bonds, then perturbed the  $x$ - and  $y$ -coordinates of each node by adding a random number drawn from a uniform distribution of width  $w$  to the  $x$ -coordinate, and a separate, independent random number drawn from the same distribution to the  $y$ -coordinate. Then, we reconnected the same node pairs— $(1, 4)$ ,  $(2, 4)$ ,  $(3, 4)$ ,  $(1, 5)$ ,  $(2, 5)$ ,  $(3, 5)$ —using rigid bonds whose lengths were now fixed to be the distances between the *perturbed* node coordinates, which yielded a perturbed network whose conformational motion produced a perturbed map  $l_{k+1} = f_p(l_k)$ .

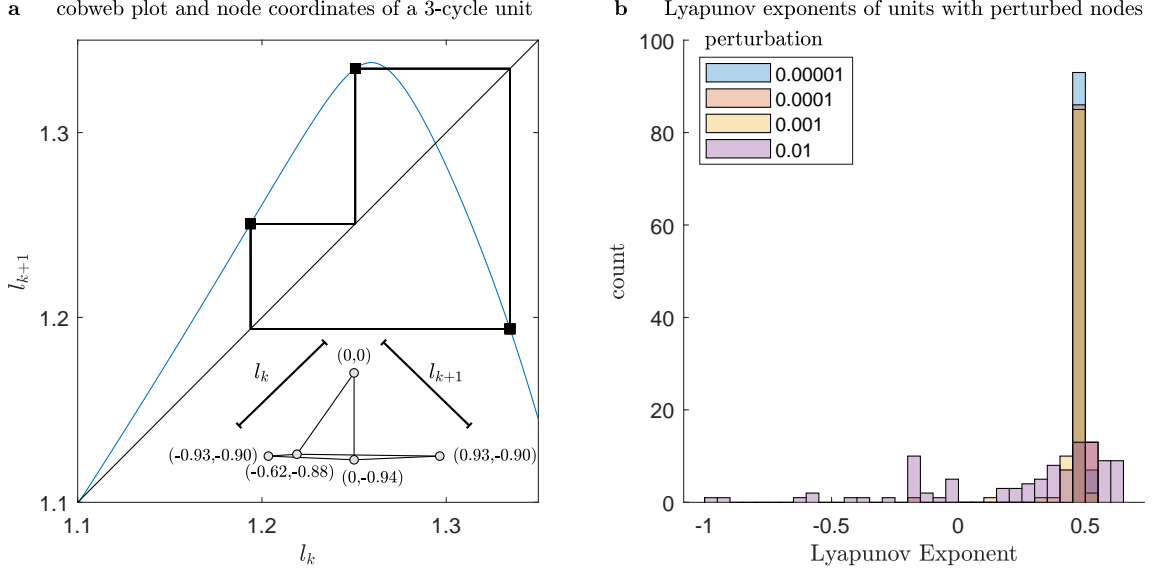


Figure 79: **Robustness of Chaos to Random Parameter Perturbations.** (left) Cobweb plot of a mechanical unit that demonstrates a 3-cycle, thereby implying chaos. The precise network used with the precise coordinates of the joints are displayed. (right) Distribution of Lyapunov exponents for random perturbations to the node coordinates of varying magnitudes.

For each width  $w \in \{0.00001, 0.0001, 0.001, 0.01\}$ , we generated 100 such perturbed units corresponding to 100 such perturbed maps, which yielded 100 corresponding Lyapunov exponents (Fig. 79, right). For perturbations  $w \in \{0.00001, 0.0001, 0.001\}$ , the property of chaos is robustly preserved, as is evidenced by the positive Lyapunov exponents. When the perturbation is  $w = 0.01$ , the property of chaos begins to fluctuate, but is still present in the majority of cases. Hence, chaos—as measured by a positive Lyapunov exponent—is preserved under a small yet finite fraction of randomly chosen bond lengths.

## CHAPTER 10 : Concluding Remarks

### 10.1. Summary of Our Process for Extracting Design Principles

In the preceding chapters, we develop several principles for engineering the microstate interactions to design advanced functions. In these concluding remarks, we summarize the overall approach used to extract these principles.

First, we define the equations that govern the evolution of the microstates,  $\mathbf{x}$ , as a function of the interactions,  $A$ . In dynamical systems, this equation takes the general form  $\dot{\mathbf{x}} = \mathbf{f}(A, \mathbf{x}, \mathbf{c})$  where  $\mathbf{c}$  generally represents extra inputs and parameters. In mechanical systems, this equation takes the general form  $\mathbf{x} = \mathbf{f}(A, \mathbf{x}, \mathbf{c})$ .

Next, we write the evolution of the microstates in an accessible algebraic form. In Chapter 2, this form involves approximating the dynamics as a linear system in Eq. 2.1 and performing an expansion of the control energy expression with respect to the connectivity matrix in Eq. 2.2. In Chapter 4, this form involves taking a differential expansion of the dynamics about a trajectory in Eq. 4.9. In Chapter 6, this form involves decomposing a mechanical network unit into a bipartite graph for which one part acts as independent variables for the design problem in Eq. 6.3. In Chapter 8, this form involves turning the nonlinear and high-dimensional design space in Eq. 8.10 into an iteration of a low-dimensional nonlinear problem in Eq. 8.12.

Finally, we use mathematical insights to use these algebraic forms for design. In Chapter 2, we find a substitution to write the control energy as an algebraic function of the connectivity matrix in Eq. 2.5. In Chapter 4, we find an approximation for writing the terms of the expansion as a function of control parameters in Eq. 4.12 and Eq. 4.13. In Chapter 6, we find a change of variables to isolate the nonlinearity into a single quadratic term in Eq. 6.4, enabling a simple and geometrically intuitive design framework. In Chapter 8, we use techniques in nonlinear dynamics to precisely design a rich set of highly complex shape changes in Eq. 8.13, 8.15, and 8.18.

## 10.2. Key Concepts for Complex Systems Engineering

### 10.2.1. *Preservation of the Microstate Interactions*

Through this process, one crucial component is to preserve as clear and direct of a relationship between the designed function and the interaction matrix  $A$  as possible. The reason for this preservation is two-fold.

First, our ability to engineer a complex system is often most straightforward at the lowest level of the interaction weights. In general, it is conceptually and practically more straightforward to modify the individual interactions than to change the eigenspectra of a matrix. For example, the surgical resection and ablation of neural tissue is performed at the level of nodes and edges, not at the level of eigenvalues. As a corollary, complex systems themselves often rely on local information at the level of interactions such as Hebbian plasticity and neighboring electrostatic interactions.

Second, there are often deep insights into complexity that are lost when moving beyond the element-by-element representation of the interactions. For example, in Chapter 2, a convenient stopping point for the derivation is Eq. 2.3, where the energetics can be written as a function of the matrix determinant. From this point, it is tempting to close by relating the determinant to the eigenvalues of a matrix. However, through careful mathematical relations, we can rewrite the energetics as a function of the connectivity weights in Eq. 2.5, which simultaneously yields a more visceral geometric intuition of network topology for control, and enables further insights into mesoscopic network organization that yields energetically favorable connectivities.

Of course, this is not to say that the *only* way to engineer complex systems is by preserving relationships between the function and the microstate interactions. We are simply providing a comment that premature abstraction, feature extraction, or coarse graining of the individual interactions may conceal interesting and important aspects of complexity design.

### 10.2.2. Nullspaces of Design are Often Immune to Complexity

Another crucial component is the utilization of *nullspaces* as design tools. One property of complex systems that makes them difficult to design is *nonlinearity*, where we do not know how the system will behave if we perturb it in a certain direction. Nullspaces offer a simple solution to this problem by mapping certain directions of perturbation to zero, such that we know precisely how the system will behave along all dimensions of a nullspace. In the design of mechanical systems, it is a serendipitous occasion that conformational motions lie in the nullspace of constraints.

For example, in Chapter 6, we make use of nullspaces in two instances: one explicit, and one implicit. In the explicit instance, we solve for the set of node motions that yields zero bond extension through the nullspace of our constraints in Eq. 6.4 and 6.7. Here, the nullspace tells us where we can place additional constraints in our network that *do not preclude* the designed motion. All other constraint placements do preclude the designed motion, and do so in a complex, nonlinear, and high-dimensional manner. However, we can ignore all such complexities, because a constraint placed *anywhere* in the nullspace will preserve the designed motion. In the implicit instance, we decompose our system into a *bipartite* network with the first part as the nodes to design, and the second part acting as variables in the design problem. By removing the bonds that connect the nodes of the second part to each other, we decouple the design problem such that the variables remain independent from each other.

This independence hints at another reason why nullspaces are often immune to complexity, which is that elements in nullspaces are independent of each other. Because nullspaces inherently map to zero, any scalar multiple or linear combination of elements in the nullspace also maps to zero. The reader may be familiar with nullspaces in the context of solutions to systems of linear equations or partial differential equations. Here, we are simply providing a comment that such a convenient tool can also be used to define nullspaces of design.

### *10.2.3. A New Twist on an Old Classic: Complex Systems Engineering Welcomes You!*

We conclude on a final note about the importance of creativity and open-mindedness in complex systems engineering. To re-iterate the words of P. W. Anderson:

“The ability to reduce everything to simple fundamental laws does not imply the ability to start from those laws and reconstruct the universe.”

Implied in these words is that construction requires us to think creatively, flexibly, and in qualitatively different ways. While it is understood that the study of complex systems is an inherently interdisciplinary endeavor, we wish to emphasize two additional points.

The first point is one of simplicity. In an era of increasingly accelerated computing, advanced optimization algorithms, large-scale simulations, and massive databases, complexity abounds in the simplest of systems, and can be fruitfully studied and designed using simple tools. Most of the methods in this dissertation are limited to a typical undergraduate curriculum involving linear and elementary algebra, calculus, mechanics, and dynamics. By no means are we saying that more advanced and sophisticated approaches are unimportant. Instead, we make the simple statement that there exist many fascinating and impactful design principles that hide in even the simplest of models. We need only to shift our perspective.

The second point accompanies the first, and is one of age. The ebb and flow of scientific trends mean that research topics gain and lose popularity through the years. However, the passage of time does not necessarily diminish the utility of a method, even in modern day and cutting edge research topics. Whether from a dusty textbook in a school library or from a recent paper in a top-tier journal, the important engineering problems of today and tomorrow do not inherently discriminate by age or popularity.

In conclusion, a fresh perspective, a unique background, and an unconventional education are necessary to make progress. So to those who wish to engineer complex systems, welcome!

## BIBLIOGRAPHY

- J. A. Acebrón, L. L. Bonilla, C. J. P. Vicente, F. Ritort, and R. Spigler. The kuramoto model: A simple paradigm for synchronization phenomena. *Reviews of modern physics*, 77(1):137, 2005.
- A. L. Alexander, J. E. Lee, M. Lazar, and A. S. Field. Diffusion tensor imaging of the brain. *Neurotherapeutics*, 4(3):316–329, 2007.
- J. Alglave, A. Fox, S. Ishtiaq, M. O. Myreen, S. Sarkar, P. Sewell, and F. Z. Nardelli. The semantics of power and ARM multiprocessor machine code. In *Proceedings of the 4th workshop on Declarative aspects of multicore programming - DAMP '09*, page 13, New York, New York, USA, 2008. ACM Press. ISBN 9781605584171. doi: 10.1145/1481839.1481842.
- N. M. Allewell. Escherichia Coli Aspartate Transcarbamoylase: Structure, Energetics, and Catalytic and Regulatory Mechanisms. *Annual Review of Biophysics and Biophysical Chemistry*, 18(1):71–92, jun 1989. ISSN 0883-9182. doi: 10.1146/annurev.bb.18.060189.000443.
- J. Alstott, M. Breakspear, P. Hagmann, L. Cammoun, and O. Sporns. Modeling the impact of lesions in the human brain. *PLoS Computational Biology*, 5(6), 2009. ISSN 1553734X. doi: 10.1371/journal.pcbi.1000408.
- P. W. Anderson. More is different. *Science*, 177(4047):393–396, 1972.
- A. A. Aristidou, K.-Y. San, and G. N. Bennett. Modification of central metabolic pathway in escherichia coli to reduce acetate accumulation by heterologous expression of the bacillus subtilis acetolactate synthase gene. *Biotechnology and Bioengineering*, 44(8):944–951, 1994. ISSN 10970290. doi: 10.1002/bit.260440810.
- L. Arnold. Stochastic differential equations. *New York*, 1974.
- L. Asimow and B. Roth. The Rigidity of Graphs. *Transactions of the American Mathematical Society*, 245:279–289, nov 1978. ISSN 00029947. doi: 10.2307/1998867.
- B. B. Avants, N. J. Tustison, G. Song, P. A. Cook, A. Klein, and J. C. Gee. A reproducible evaluation of ANTs similarity metric performance in brain image registration. *NeuroImage*, 2011.
- A. Avena-Koenigsberger, B. Misic, and O. Sporns. Communication dynamics in complex brain networks. *Nature Reviews Neuroscience*, 19(1):17–33, 2018.
- L. Avery and Y.-J. You. C. elegans feeding. *WormBook: The Online Review of C. elegans Biology [Internet]*, 2018.
- M. Balcerzak, D. Pikunov, and A. Dabrowski. The fastest, simplified method of lyapunov

- exponents spectrum estimation for continuous-time dynamical systems. *Nonlinear Dynamics*, 94(4):3053–3065, 2018.
- A. L. Barabasi, N. Gulbahce, and J. Loscalzo. Network medicine: a network-based approach to human disease. *Nat Rev Genet*, 12(1):56–68, 2011.
- C. I. Bargmann. Chemosensation in *c. elegans*. *WormBook: The online review of C. elegans biology [Internet]*, 2006.
- A. Baskar and S. Bandyopadhyay. An algorithm to compute the finite roots of large systems of polynomial equations arising in kinematic synthesis. *Mechanism and Machine Theory*, 133:493–513, 2019.
- D. S. Bassett and E. Bullmore. Small-world brain networks revisited. *Neuroscientist*, In Press, 2016.
- D. S. Bassett and O. Sporns. Network neuroscience. *Nature Neuroscience*, In Press, 2016.
- D. S. Bassett and O. Sporns. Network neuroscience. *Nature neuroscience*, 20(3):353–364, 2017.
- D. S. Bassett, J. A. Brown, V. Deshpande, J. M. Carlson, and S. T. Grafton. Conserved and variable architecture of human white matter connectivity. *Neuroimage*, 54(2):1262–1279, 2011.
- D. S. Bassett, E. T. Owens, K. E. Daniels, and M. A. Porter. Influence of network topology on sound propagation in granular materials. *Phys Rev E*, 86(4 Pt 1):041306, 2012.
- D. S. Bassett, N. F. Wymbs, M. A. Porter, P. J. Mucha, and S. T. Grafton. Cross-linked structure of network evolution. *Chaos*, 24(1):013112, 2014.
- D. S. Bassett, A. N. Khambhati, and S. T. Grafton. Emerging frontiers of neuroengineering: A network science of brain connectivity. *Annual Reviews in Biomedical Engineering*, Under Consideration, 2017.
- T. Bekolay, J. Bergstra, E. Hunsberger, T. DeWolf, T. C. Stewart, D. Rasmussen, X. Choo, A. Voelker, and C. Eliasmith. Nengo: a python tool for building large-scale functional brain models. *Frontiers in neuroinformatics*, 7:48, 2014.
- J. M. Bernabei, O. Owoputi, S. D. Small, N. T. Nyema, E. Dumenyo, J. Kim, S. N. Baldassano, C. Painter, E. C. Conrad, T. M. Ganguly, et al. A full-stack application for detecting seizures and reducing data during continuous electroencephalogram monitoring. *Critical Care Explorations*, 3(7), 2021.
- K. Bertoldi, V. Vitelli, J. Christensen, and M. van Hecke. Flexible mechanical metamaterials. *Nature Reviews Materials*, 2:17066, oct 2017a. ISSN 2058-8437. doi: 10.1038/natrevmats.2017.66.



- K. Bertoldi, V. Vitelli, J. Christensen, and M. Van Hecke. Flexible mechanical metamaterials. *Nature Reviews Materials*, 2(11):1–11, 2017b.
- L. M. Bettencourt, G. J. Stephens, M. I. Ham, and G. W. Gross. Functional structure of cortical neuronal networks grown in vitro. *Phys Rev E Stat Nonlin Soft Matter Phys*, 75(2 Pt 1):021915, 2007.
- R. F. Betzel, S. Gu, J. D. Medaglia, F. Pasqualetti, and D. S. Bassett. Optimally controlling the human connectome: the role of network topology. *Scientific Reports*, 6(1):30770, nov 2016a. ISSN 2045-2322. doi: 10.1038/srep30770.
- R. F. Betzel, S. Gu, J. D. Medaglia, F. Pasqualetti, and D. S. Bassett. Optimally controlling the human connectome: the role of network topology. *Scientific reports*, 6(1):1–14, 2016b.
- E. Bolker and B. Roth. When is a bipartite graph a rigid framework? *Pacific Journal of Mathematics*, 90(1):27–44, sep 1980. ISSN 0030-8730. doi: 10.2140/pjm.1980.90.27.
- B. Bollobas. *Random Graphs*. Academic Press, 1985.
- R. W. Brockett. Volterra series and geometric control theory. *Automatica*, 12(2):167–176, 1976.
- S. L. Brunton, B. W. Brunton, J. L. Proctor, and J. N. Kutz. Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. *PloS one*, 11(2):e0150171, 2016.
- M. Budišić, R. Mohr, and I. Mezić. Applied koopmanism. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 22(4):047510, 2012.
- Y. Burak and I. R. Fiete. Accurate Path Integration in Continuous Attractor Network Models of Grid Cells. *PLoS Computational Biology*, 5(2):e1000291, feb 2009. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1000291.
- M. Burrows and G. Sutton. Interacting gears synchronize propulsive leg movements in a jumping insect. *Science*, 341(6151):1254–1256, sep 2013. ISSN 0036-8075. doi: 10.1126/science.1240284.
- K. Caeyenberghs, H. Verhelst, A. Clemente, and P. H. Wilson. Mapping the functional connectome in traumatic brain injury: What can graph metrics tell us? *Neuroimage*, S1053-8119(16):30694–30692, 2016.
- C. R. Calladine. Buckminster Fuller’s “Tensegrity” structures and Clerk Maxwell’s rules for the construction of stiff frames. *International Journal of Solids and Structures*, 14(2):161–172, 1978. ISSN 00207683. doi: 10.1016/0020-7683(78)90052-5.
- J. M. Carroll. Letter knowledge precipitates phoneme segmentation, but not phoneme invariance. *Journal of Research in Reading*, 27(3):212–225, aug 2004. ISSN 0141-0423. doi: 10.1111/j.1467-9817.2004.00228.x.

- J. R. Cash and A. H. Karp. A variable order runge-kutta method for initial value problems with rapidly varying right-hand sides. *ACM Transactions on Mathematical Software (TOMS)*, 16(3):201–222, 1990.
- J.-P. Changeux and S. J. Edelstein. Allosteric Receptors after 30 Years. *Neuron*, 21(5): 959–980, nov 1998. ISSN 08966273. doi: 10.1016/S0896-6273(00)80616-9.
- B. G.-g. Chen, B. Liu, A. A. Evans, J. Paulose, I. Cohen, V. Vitelli, and C. D. Santangelo. Topological mechanics of origami and kirigami. *Physical Review Letters*, 116(13):135501, mar 2016. ISSN 0031-9007. doi: 10.1103/PhysRevLett.116.135501.
- H. I. Chen, M. Attiah, G. Baltuch, D. H. Smith, R. H. Hamilton, and T. H. Lucas. Harnessing plasticity for the treatment of neurosurgical disorders: an overview. *World Neurosurg*, 82(5):648–659, 2014.
- T. Chen, M. Pauly, and P. M. Reis. A reprogrammable mechanical metamaterial with stable memory. *Nature*, 589(7842):386–390, 2021.
- K. C. Cheung and N. Gershenfeld. Reversibly assembled cellular composite materials. *Science*, 341(6151):1219–1221, sep 2013. ISSN 0036-8075. doi: 10.1126/science.1240889.
- A. S. Chiang, C. Y. Lin, C. C. Chuang, H. M. Chang, C. H. Hsieh, C. W. Yeh, C. T. Shih, J. J. Wu, G. T. Wang, Y. C. Chen, C. C. Wu, G. Y. Chen, Y. T. Ching, P. C. Lee, C. Y. Lin, H. H. Lin, C. C. Wu, H. W. Hsu, Y. A. Huang, J. Y. Chen, H. J. Chiang, C. F. Lu, R. F. Ni, C. Y. Yeh, and J. K. Hwang. Three-dimensional reconstruction of brain-wide wiring networks in *Drosophila* at single-cell resolution. *Current Biology*, 21(1):1–11, 2011.
- S. Ching, E. N. Brown, and M. A. Kramer. Distributed control in a mean-field cortical network model: implications for seizure suppression. *Phys Rev E Stat Nonlin Soft Matter Phys*, 86(2 Pt 1):021920, 2012.
- G. P. Choi, L. H. Dudte, and L. Mahadevan. Programming shape using kirigami tessellations. *Nature materials*, 18(9):999–1004, 2019.
- E. G. Chrysikou and R. H. Hamilton. Noninvasive brain stimulation in the treatment of aphasia: exploring interhemispheric relationships and their implications for neurorehabilitation. *Restor Neurol Neurosci*, 29(6):375–394, 2011.
- M. Cieslak and S. T. Grafton. Local termination pattern analysis: a tool for comparing white matter morphology. *Brain Imaging Behav*, 8(2):292–299, 2014.
- A. Clauset, C. Moore, and M. E. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453(7191):98–101, 2008.
- G. M. Cockrell, Y. Zheng, W. Guo, A. W. Peterson, J. K. Truong, and E. R. Kantrowitz. New Paradigm for Allosteric Regulation of *Escherichia coli* Aspartate Transcarbamoylase. *Biochemistry*, 52(45):8036–8047, nov 2013. ISSN 0006-2960. doi: 10.1021/bi401205n.

- R. Connelly and J.-M. Schlenker. On the infinitesimal rigidity of weakly convex polyhedra. *European Journal of Combinatorics*, 31(4):1080–1090, 2010.
- S. P. Cornelius, W. L. Kath, and A. E. Motter. Realistic control of network dynamics. *Nature communications*, 4:1942, 2013a. ISSN 2041-1723. doi: 10.1038/ncomms2939.
- S. P. Cornelius, W. L. Kath, and A. E. Motter. Realistic control of network dynamics. *Nature communications*, 4(1):1–9, 2013b.
- C. Coulais, D. Sounas, and A. Alù. Static non-reciprocity in mechanical metamaterials. *Nature*, 542(7642):461–464, feb 2017. ISSN 0028-0836. doi: 10.1038/nature21044.
- S. M. Courtney, L. G. Ungerleider, K. Keil, and J. V. Haxby. Transient and sustained activity in a distributed neural system for human working memory. *Nature*, 386(6625):608–611, 1997.
- F. I. Craik and E. Bialystok. Cognition through the lifespan: mechanisms of change. *Trends in Cognitive Sciences*, 10(3):131–138, mar 2006. ISSN 13646613. doi: 10.1016/j.tics.2006.01.007.
- H. Crapo. Structural rigidity. *Structural Topology*, 1:26–45, apr 1979a. ISSN 8750-7587.
- H. Crapo. Structural rigidity. *Structural Topology*, 73(1):26–45, 1979b.
- T. E. Creighton. *Proteins: structures and molecular properties*. Macmillan, 1993.
- H. Cui, R. Hensleigh, D. Yao, D. Maurya, P. Kumar, M. G. Kang, S. Priya, and X. Zheng. Three-dimensional printing of piezoelectric materials with designed anisotropy and directional response. *Nature Materials*, 18(3):234–241, mar 2019. ISSN 1476-1122. doi: 10.1038/s41563-018-0268-1.
- S. A. Cummer, J. Christensen, and A. Alù. Controlling sound with acoustic metamaterials. *Nature Reviews Materials*, 1(3):16001, mar 2016. ISSN 2058-8437. doi: 10.1038/natrevmats.2016.1.
- J. Dagdelen, J. Montoya, M. de Jong, and K. Persson. Computational prediction of new auxetic materials. *Nature Communications*, 8(1):323, dec 2017. ISSN 2041-1723. doi: 10.1038/s41467-017-00399-6.
- C. Detweiler, M. Vona, Y. Yoon, Seung-Kook Yun, and D. Rus. Self-assembling mobile linkages. *IEEE Robotics & Automation Magazine*, 14(4):45–55, dec 2007. ISSN 1070-9932. doi: 10.1109/M-RA.2007.908971.
- G. F. Donnay, S. K. Rankin, M. Lopez-Gonzalez, P. Jiradejvong, and C. J. Limb. Neural Substrates of Interactive Musical Improvisation: An fMRI Study of ‘Trading Fours’ in Jazz. *PLoS ONE*, 9(2):e88665, feb 2014. ISSN 1932-6203. doi: 10.1371/journal.pone.0088665.

- J. S. Duncan, J. W. Sander, S. M. Sisodiya, and M. C. Walker. Adult epilepsy. *The Lancet*, 367(9516):1087–1100, 2006.
- F. A. Dunn and R. O. L. Wong. Diverse Strategies Engaged in Establishing Stereotypic Wiring Patterns among Neurons Sharing a Common Input at the Visual System’s First Synapse. *Journal of Neuroscience*, 32(30):10306–10317, jul 2012. ISSN 0270-6474. doi: 10.1523/JNEUROSCI.1581-12.2012.
- V. M. Eguíluz, N. Masuda, and J. Fernández-Gracia. Bayesian decision making in human collectives with binary choices. *PLoS One*, 10(4):e0121332, 2015.
- C. Eliasmith and C. H. Anderson. *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. MIT press, 2003.
- E. Estrada and N. Hatano. Communicability in complex networks. *Phys Rev E Stat Nonlin Soft Matter Phys*, 77(3 Pt 2):036111, 2008.
- R. L. Faulkner, M.-H. Jang, X.-B. Liu, X. Duan, K. A. Sailor, J. Y. Kim, S. Ge, E. G. Jones, G.-l. Ming, H. Song, and H.-J. Cheng. Development of hippocampal mossy fiber synaptic outputs by new neurons in the adult brain. *Proceedings of the National Academy of Sciences*, 105(37):14157–14162, sep 2008. ISSN 0027-8424. doi: 10.1073/pnas.0806658105.
- T. Feagin. A tenth-order runge-kutta method with error estimate. In *Proceedings of the IAENG Conference on Scientific Computing*, 2007.
- M. S. Fee and C. Scharff. The Songbird as a Model for the Generation and Learning of Complex Sequential Behaviors. *ILAR Journal*, 51(4):362–377, jan 2010. ISSN 1084-2020. doi: 10.1093/ilar.51.4.362.
- L. A. Feldkamp, G. Puskorius, and P. Moore. Adaptive behavior from fixed weight networks. *Information Sciences*, 98(1):217 – 235, 1997. ISSN 0020-0255. doi: [https://doi.org/10.1016/S0020-0255\(96\)00216-2](https://doi.org/10.1016/S0020-0255(96)00216-2).
- G. R. Fernandez. On how network architecture determines the dominant patterns of spontaneous neural activity. *PLoS One*, 3(5):e2148, 2008.
- R. FitzHugh. Impulses and physiological states in theoretical models of nerve membrane. *Biophysical journal*, 1(6):445–466, 1961.
- H. Flechsig. Design of elastic networks with evolutionary optimized long-range communication as mechanical models of allosteric proteins. *Biophysical Journal*, 113(3):558–571, aug 2017. ISSN 00063495. doi: 10.1016/j.bpj.2017.06.043.
- B. Fu, E. Sperber, and F. Eke. Solar sail technology—A state of the art review. *Progress in Aerospace Sciences*, 86:1–19, oct 2016a. ISSN 03760421. doi: 10.1016/j.paerosci.2016.07.001.

- B. Fu, E. Sperber, and F. Eke. Solar sail technology—a state of the art review. *Progress in Aerospace Sciences*, 86:1–19, 2016b.
- A. Gierer and H. Meinhardt. A theory of biological pattern formation. *Kybernetik*, 12(1): 30–39, 1972.
- C. Giusti, L. Papadopoulos, E. T. Owens, K. E. Daniels, and D. S. Bassett. Topological and geometric measurements of force-chain structure. *Physical Review E*, 94(3):032909, sep 2016. ISSN 2470-0045. doi: 10.1103/PhysRevE.94.032909.
- J. I. Gold and M. N. Shadlen. The Neural Basis of Decision Making. *Annual Review of Neuroscience*, 30(1):535–574, jul 2007. ISSN 0147-006X. doi: 10.1146/annurev.neuro.29.051605.113038.
- T. Gonen, T. Gazit, A. Korn, A. Kirschner, D. Perry, T. Hendler, and Z. Ram. Intra-operative multi-site stimulation: Expanding methodology for cortical brain mapping of language functions. *PLOS ONE*, 12(7):e0180740, jul 2017. ISSN 1932-6203. doi: 10.1371/journal.pone.0180740.
- C. P. Goodrich, A. J. Liu, and S. R. Nagel. The Principle of Independent Bond-Level Response: Tuning by Pruning to Exploit Disorder for Global Behavior. *Physical Review Letters*, 114(22):225501, jun 2015. ISSN 0031-9007. doi: 10.1103/PhysRevLett.114.225501.
- K. Gorgolewski, C. D. Burns, C. Madison, D. Clark, Y. O. Halchenko, M. L. Waskom, and S. S. Ghosh. Nipype: a flexible, lightweight and extensible neuroimaging data processing framework in python. *Frontiers in neuroinformatics*, 2011.
- A. Graves, G. Wayne, M. Reynolds, T. Harley, I. Danihelka, A. Grabska-Barwińska, S. G. Colmenarejo, E. Grefenstette, T. Ramalho, J. Agapiou, A. P. Badia, K. M. Hermann, Y. Zwols, G. Ostrovski, A. Cain, H. King, C. Summerfield, P. Blunsom, K. Kavukcuoglu, and D. Hassabis. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476, oct 2016. ISSN 0028-0836. doi: 10.1038/nature20101.
- K. Gregor, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra. Draw: A recurrent neural network for image generation. In *International Conference on Machine Learning*, pages 1462–1471. PMLR, 2015.
- A. Griffa, P. S. Baumann, J.-P. Thiran, and P. Hagmann. Structural connectomics in brain diseases. *NeuroImage*, 80:515–526, Oct. 2013.
- H. Grimm and B. Dorner. On the mechanism of the  $\alpha$ - $\beta$  phase transformation of quartz. *Journal of Physics and Chemistry of Solids*, 36(5):407–413, may 1975. ISSN 00223697. doi: 10.1016/0022-3697(75)90066-9.
- S. Gu, F. Pasqualetti, M. Cieslak, Q. K. Telesford, A. B. Yu, A. E. Kahn, J. D. Medaglia, J. M. Vettel, M. B. Miller, S. T. Grafton, and D. S. Bassett. Controllability of structural

- brain networks. *Nature Communications*, 6:8414, 2015. ISSN 2041-1723. doi: 10.1038/ncomms9414.
- S. Gu, R. F. Betzel, M. G. Mattar, M. Cieslak, P. R. Delio, S. T. Grafton, F. Pasqualetti, and D. S. Bassett. Optimal trajectories of brain state transitions. *Neuroimage*, In Press, 2017.
- S. Guest. The stiffness of prestressed frameworks: A unifying approach. *International Journal of Solids and Structures*, 43(3-4):842–854, feb 2006. ISSN 00207683. doi: 10.1016/j.ijsolstr.2005.03.008.
- J. Guo and H. X. Zhou. Protein allostery and conformational dynamics. *Chem Rev*, 116(11):6503–6515, 2016.
- P. Hagmann, L. Cammoun, X. Gigandet, R. Meuli, C. J. Honey, V. J. Wedeen, and O. Sporns. Mapping the structural core of human cerebral cortex. *PLoS biology*, 6(7):e159, July 2008.
- R. Hartenberg and J. Danavit. *Kinematic synthesis of linkages*. New York: McGraw-Hill, 1964.
- M. Hegarty. Mechanical reasoning by mental simulation. *Trends in Cognitive Sciences*, 8(6):280–285, jun 2004. ISSN 13646613. doi: 10.1016/j.tics.2004.04.001.
- S. Herculano-Houzel. The human brain in numbers: a linearly scaled-up primate brain. *Frontiers in human neuroscience*, 3:31, 2009.
- J. P. Hespanha. *Linear systems theory*. Princeton university press, 2018.
- A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500–544, 1952.
- C. J. Honey, O. Sporns, L. Cammoun, X. Gigandet, J. P. Thiran, R. Meuli, and P. Hagmann. Predicting human resting-state functional connectivity from structural connectivity. *Proc Natl Acad Sci U S A*, 106(6):2035–2040, 2009.
- J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.
- D.-G. Hwang and M. D. Bartlett. Tunable mechanical metamaterials through hybrid kirigami structures. *Scientific reports*, 8(1):1–8, 2018.
- K. Hwang, M. A. Bertolero, W. B. Liu, and M. D’Esposito. The human thalamus is an integrative hub for functional brain networks. *Journal of Neuroscience*, 37(23):5594–5607, 2017.

- P. J. Ifft, S. Shokur, Z. Li, M. A. Lebedev, and M. A. L. Nicolelis. A Brain-Machine Interface Enables Bimanual Arm Movements in Monkeys. *Science Translational Medicine*, 5(210): 210ra154–210ra154, nov 2013. ISSN 1946-6234. doi: 10.1126/scitranslmed.3006159.
- Y. Ilyashenko. Centennial history of hilbert’s 16th problem. *Bulletin of the American Mathematical Society*, 39(3):301–354, 2002.
- J. A. Jackson, M. C. Messner, N. A. Dudukovic, W. L. Smith, L. Bekker, B. Moran, A. M. Golobic, A. J. Pascall, E. B. Duoss, K. J. Loh, and C. M. Spadaccini. Field responsive mechanical metamaterials. *Science advances*, 4(12):eaau6419, 2018.
- D. J. Jacobs and M. F. Thorpe. Generic rigidity percolation: The pebble game. *Physical Review Letters*, 75(22):4051–4054, nov 1995. ISSN 0031-9007. doi: 10.1103/PhysRevLett.75.4051.
- H. Jaeger. The “echo state” approach to analysing and training recurrent neural networks – with an Erratum note. *GMD Report*, 1(148):1–47, dec 2010.
- T. A. Jarrell, Y. Wang, A. E. Bloniarz, C. A. Brittin, M. Xu, J. N. Thomson, D. G. Albertson, D. H. Hall, and S. W. Emmons. The Connectome of a Decision-Making Neural Network. *Science*, 337(6093):437–444, jul 2012. ISSN 0036-8075. doi: 10.1126/science.1221762.
- E. T. Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.
- L. Jin, A. E. Forte, B. Deng, A. Rafsanjani, and K. Bertoldi. Kirigami-inspired inflatables with programmable shapes. *Advanced Materials*, 32(33):2001863, 2020.
- M. D. Johnson, H. H. Lim, T. Netoff, A. T. Connolly, N. Johnson, A. Roy, A. Holt, K. O. Lim, J. R. Carey, J. L. Vitek, and B. He. Neuromodulation for brain disorders: challenges and opportunities. *IEEE Trans Biomed Eng*, 60(3):610–624, 2013.
- T. Kailath. *Linear Systems*. Prentice-Hall, 1980.
- R. E. Kalman. Mathematical description of linear dynamical systems. *J. SIAM Control Ser. A*, 1(1963):152–192, 1963.
- C. L. Kane and T. C. Lubensky. Topological boundary modes in isostatic lattices. *Nature Physics*, 10(1):39–45, jan 2014. ISSN 1745-2473. doi: 10.1038/nphys2835.
- G. Karlebach and R. Shamir. Modelling and analysis of gene regulatory networks. *Nature reviews Molecular cell biology*, 9(10):770–780, 2008.
- A. B. Kempe. On a General Method of describing Plane Curves of the  $n$  th degree by Linkwork. *Proceedings of the London Mathematical Society*, s1-7(1):213–216, nov 1875. ISSN 00246115. doi: 10.1112/plms/s1-7.1.213.

- A. N. Khambhati, K. Davis, T. Lucas, B. Litt, and D. S. Bassett. Virtual cortical resection reveals push-pull network control preceding seizure evolution. *Neuron*, In Press, 2016.
- J. Z. Kim and D. S. Bassett. Linear dynamics and control of brain networks. In *Neural Engineering*, pages 497–518. Springer, 2020.
- J. Z. Kim, J. M. Soffer, A. E. Kahn, J. M. Vettel, F. Pasqualetti, and D. S. Bassett. Role of graph architecture in controlling dynamical networks with applications to neural systems. *Nature physics*, 14(1):91–98, 2018.
- J. Z. Kim, Z. Lu, and D. S. Bassett. Design of large sequential conformational change in mechanical networks. *arXiv preprint arXiv:1906.08400 (Accepted, PRX)*, 2019a.
- J. Z. Kim, Z. Lu, S. H. Strogatz, and D. S. Bassett. Conformational control of mechanical networks. *Nature Physics*, 15(7):714–720, 2019b.
- J. Z. Kim, Z. Lu, E. Nozari, G. J. Pappas, and D. S. Bassett. Teaching recurrent neural networks to infer global temporal structure from local examples. *Nature Machine Intelligence*, 3(4):316–323, 2021.
- L. G. Kini, J. M. Bernabei, F. Mikhail, P. Hadar, P. Shah, A. N. Khambhati, K. Oechsel, R. Archer, J. Boccanfuso, E. Conrad, et al. Virtual resection predicts surgical outcome for drug-resistant epilepsy. *Brain*, 142(12):3892–3905, 2019.
- F. Klimm, D. S. Bassett, J. M. Carlson, and P. J. Mucha. Resolving structural variability in network models and the brain. *PLoS Comput Biol*, 10(3):e1003491, 2014.
- C. Klos, Y. F. K. Kossio, S. Goedeke, A. Gilra, and R.-M. Memmesheimer. Dynamical learning of dynamics. *Physical Review Letters*, 125(8):088103, 2020.
- H. M. Kolken and A. A. Zadpoor. Auxetic mechanical metamaterials. *RSC advances*, 7(9):5111–5129, 2017.
- H. M. Kolken, S. Janbaz, S. M. Leeflang, K. Lietaert, H. H. Weinans, and A. A. Zadpoor. Rationally designed meta-implants: a combination of auxetic and conventional meta-biomaterials. *Materials Horizons*, 5(1):28–35, 2018.
- C. Körner and Y. Liebold-Ribeiro. A systematic approach to identify cellular auxetic materials. *Smart Materials and Structures*, 24(2):025013, feb 2015. ISSN 0964-1726. doi: 10.1088/0964-1726/24/2/025013.
- Y. Kuang. *Delay differential equations*. University of California Press, 2012.
- J. R. Kubricht, K. J. Holyoak, and H. Lu. Intuitive Physics: Current Research and Controversies. *Trends in Cognitive Sciences*, 21(10):749–759, oct 2017. ISSN 13646613. doi: 10.1016/j.tics.2017.06.002.



- S. Kumar, I. Dasgupta, J. D. Cohen, N. D. Daw, and T. L. Griffiths. Meta-learning of compositional task distributions in humans and machines, 2020.
- P. Kwan, A. Arzimanoglou, A. T. Berg, M. J. Brodie, W. Allen Hauser, G. Mathern, S. L. Moshé, E. Perucca, S. Wiebe, and J. French. Definition of drug resistant epilepsy: consensus proposal by the ad hoc task force of the ilae commission on therapeutic strategies, 2010.
- G. Langer and U. Parlitz. Modeling parameter dependence from time series. *Physical Review E*, 70(5):056217, 2004.
- L. Lanteaume, S. Khalfa, J. Régis, P. Marquis, P. Chauvel, and F. Bartolomei. Emotion induction after direct intracerebral stimulations of human amygdala. *Cerebral Cortex*, 17:1307–1313, 2007.
- Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.
- J. Lee and I. Tashev. High-level feature representation using recurrent neural network for speech emotion recognition. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, volume 2015-Janua, pages 1537–1540, 2015.
- J.-H. Lee, J. P. Singer, and E. L. Thomas. Micro-/Nanostructured Mechanical Metamaterials. *Advanced Materials*, 24(36):4782–4810, sep 2012.
- M. H. Lee. Analytical study of the superstable 3-cycle in the logistic map. *Journal of Mathematical Physics*, 50(12):122702, 2009. doi: 10.1063/1.3266875.
- A. Leemans and D. K. Jones. The B-matrix must be rotated when correcting for subject motion in DTI data. *Magnetic resonance in medicine*, 61(6):1336–1349, June 2009.
- A. L. Lehninger, D. L. Nelson, M. M. Cox, M. M. Cox, et al. *Lehninger principles of biochemistry*. Macmillan, 2005.
- J. K. Leman, B. D. Weitzner, S. M. Lewis, J. Adolf-Bryfogle, N. Alam, R. F. Alford, M. Aprahamian, D. Baker, K. A. Barlow, P. Barth, et al. Macromolecular modeling and design in rosetta: recent methods and frameworks. *Nature methods*, 17(7):665–680, 2020.
- A. Lendlein, H. Jiang, O. Jünger, and R. Langer. Light-induced shape-memory polymers. *Nature*, 434(7035):879–882, 2005.
- C. Li and J. C. Sprott. An infinite 3-d quasiperiodic lattice of chaotic attractors. *Physics Letters A*, 382(8):581–587, 2018.
- J. Li, J. Guo, X. Ou, M. Zhang, Y. Li, and Z. Liu. Mechanical coupling of the multiple structural elements of the large-conductance mechanosensitive channel during expansion. *Proceedings of the National Academy of Sciences*, 112(34):10726–10731, aug 2015. ISSN 0027-8424. doi: 10.1073/pnas.1503202112.

- T.-Y. Li and J. A. Yorke. Period three implies chaos. In *The theory of chaotic attractors*, pages 77–84. Springer, 2004.
- C. T. Lin. Structural controllability. *IEEE Trans. Auto. Control*, AC-19:201–208, 1974.
- G. P. Lisi and J. P. Loria. Allosteric in enzyme catalysis. *Current opinion in structural biology*, 47:123–130, 2017.
- B. Liu, J. L. Silverberg, A. A. Evans, C. D. Santangelo, R. J. Lang, T. C. Hull, and I. Cohen. Topological kinematics of origami metamaterials. *Nature Physics*, 14(8):811–815, aug 2018. ISSN 1745-2473. doi: 10.1038/s41567-018-0150-8.
- T.-W. Liu and F. Semperlotti. Tunable acoustic Valley–Hall edge states in reconfigurable phononic elastic waveguides. *Physical Review Applied*, 9(1):014001, jan 2018. ISSN 2331-7019. doi: 10.1103/PhysRevApplied.9.014001.
- Y. Y. Liu, J. J. Slotine, and A. L. Barabasi. Controllability of complex networks. *Nature*, 473(7346):167–173, 2011.
- Z. Liu, S. Lin, N. Deng, D. P. McGovern, and S. Piantadosi. Sparse inverse covariance estimation with l0 penalty for network construction with omics data. *J Comput Biol*, 23(3):192–202, 2016.
- E. N. Lorenz. Deterministic Nonperiodic Flow. *Journal of the Atmospheric Sciences*, 20(2):130–141, mar 1963. ISSN 0022-4928.
- Z. Lu and D. S. Bassett. Invertible generalized synchronization: A putative mechanism for implicit learning in neural systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 30(6):063133, 2020.
- L. A. Lubbers and M. van Hecke. Excess floppy modes and multibranch mechanisms in metamaterials with symmetries. *Physical Review E*, 100(2):021001, 2019.
- J. A. Lukin and C. Ho. The Structure-Function Relationship of Hemoglobin in Solution at Atomic Resolution. *Chemical Reviews*, 104(3):1219–1230, mar 2004. ISSN 0009-2665. doi: 10.1021/cr940325w.
- J. A. Lukin, G. Kontaxis, V. Simplaceanu, Y. Yuan, A. Bax, and C. Ho. Quaternary structure of hemoglobin in solution. *Proceedings of the National Academy of Sciences*, 100(2):517–520, 2003.
- A. I. Lurie. *Analytical mechanics*. Springer Science & Business Media, 2002.
- C. P. Macol, H. Tsuruta, B. Stec, and E. R. Kantrowitz. Direct structural evidence for a concerted allosteric transition in *Escherichia coli* aspartate transcarbamoylase. *Nature Structural Biology*, 8(5):423–6, may 2001a. ISSN 1072-8368. doi: 10.1038/87582.

- C. P. Macol, H. Tsuruta, B. Stec, and E. R. Kantrowitz. Direct structural evidence for a concerted allosteric transition in *Escherichia coli* aspartate transcarbamoylase. *Nature structural biology*, 8(5):423–426, may 2001b. ISSN 1072-8368. doi: 10.1038/87582.
- B. B. Mandelbrot and B. B. Mandelbrot. *The fractal geometry of nature*, volume 1. WH freeman New York, 1982.
- A. V. Mantzaris, D. S. Bassett, N. F. Wymbs, E. Estrada, M. A. Porter, P. J. Mucha, S. T. Grafton, and D. J. Higham. Dynamic network centrality summarizes learning in the human brain. *Journal of Complex Networks*, 1(1):83–92, 2013.
- X. Mao and T. C. Lubensky. Maxwell lattices and topological mechanics. *Annual Review of Condensed Matter Physics*, 9(1):413–433, mar 2018. ISSN 1947-5454. doi: 10.1146/annurev-conmatphys-033117-054235.
- F. Martini. *Anatomy and physiology*’2007 ed. 2007 edition. *Rex Bookstore, Inc*, 2007.
- A. M. Mathai and S. B. Provost. *Quadratic Forms in Random Variables*. CRC Press, 1 edition, 1992. ISBN 978-0824786915.
- J. C. Maxwell. On the calculation of the equilibrium and stiffness of frames. *Philosophical Magazine Series 4*, 27(182):294–299, 1864a. ISSN 1941-5982. doi: 10.1080/14786446408643668.
- J. C. Maxwell. On the calculation of the equilibrium and stiffness of frames. *Philosophical Magazine Series 4*, 27(182):294–299, 1864b. ISSN 1941-5982. doi: 10.1080/14786446408643668.
- T. J. McCabe. A complexity measure. *IEEE Transactions on software Engineering*, (4):308–320, 1976.
- J. M. McCarthy and G. S. Soh. *Geometric design of linkages*, volume 11. Springer Science & Business Media, 2010.
- W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.
- D. Melancon, B. Gorissen, C. J. García-Mora, C. Hoberman, and K. Bertoldi. Multistable inflatable origami structures at the metre scale. *Nature*, 592(7855):545–550, 2021.
- C. D. Meyer. *Matrix Analysis and Applied Linear Algebra*. SIAM, 2001. ISBN 0898714540.
- C. Micheletti, P. Carloni, and A. Maritan. Accurate and efficient description of protein vibrational dynamics: comparing molecular dynamics and gaussian models. *Proteins: Structure, Function, and Bioinformatics*, 55(3):635–645, 2004.
- T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur. Recurrent neural

- network based language model. In *Interspeech*, volume 2, pages 1045–1048. Makuhari, 2010.
- S. K. Mohanty and V. Lakshminarayanan. Optical techniques in optogenetics. *Journal of Modern Optics*, 62(12):949–970, jul 2015. ISSN 0950-0340. doi: 10.1080/09500340.2015.1010620.
- K. Mori and T. Saito. Effects of stent structure on stent flexibility measurements. *Annals of Biomedical Engineering*, 33(6):733–742, jun 2005. ISSN 0090-6964. doi: 10.1007/s10439-005-2807-6.
- E. I. Moser, E. Kropff, and M.-B. Moser. Place Cells, Grid Cells, and the Brain’s Spatial Representation System. *Annual Review of Neuroscience*, 31(1):69–89, jul 2008. ISSN 0147-006X. doi: 10.1146/annurev.neuro.31.061307.090723.
- A. E. Motter. Networkcontology. *Chaos*, 25(9):097621, 2015.
- S. F. Muldoon, F. Pasqualetti, S. Gu, M. Cieslak, S. T. Grafton, J. M. Vettel, and D. S. Bassett. Stimulation-based control of dynamic brain networks. *PLoS Comp Biol*, In Press, 2016.
- S. Nansai, M. R. Elara, and M. Iwase. Dynamic analysis and modeling of jansen mechanism. *Procedia Engineering*, 64:1562 – 1571, 2013. ISSN 1877-7058. doi: <https://doi.org/10.1016/j.proeng.2013.09.238>. International Conference on Design and Manufacturing (IConDM2013).
- M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45: 167–256, 2003.
- M. E. J. Newman. *Networks: An Introduction*. Oxford University Press, 2010.
- H. T. Odum. Self-organization, transformity, and information. *Science*, 242(4882):1132–1139, 1988.
- S. W. Oh, J. A. Harris, L. Ng, B. Winslow, N. Cain, S. Mihalas, Q. Wang, C. Lau, L. Kuan, A. M. Henry, M. T. Mortrud, B. Ouellette, T. N. Nguyen, S. A. Sorensen, C. R. Slaughterbeck, W. Wakeman, Y. Li, D. Feng, A. Ho, E. Nicholas, K. E. Hirokawa, P. Bohn, K. M. Joines, H. Peng, M. J. Hawrylycz, J. W. Phillips, J. G. Hohmann, P. Wohnoutka, C. R. Gerfen, C. Koch, A. Bernard, C. Dang, A. R. Jones, and H. Zeng. A mesoscale connectome of the mouse brain. *Nature*, 508(7495):207–214, 2014.
- E. Ott, C. Grebogi, and J. A. Yorke. Controlling chaos. *Physical Review Letters*, 64(11): 1196–1199, mar 1990. ISSN 0031-9007. doi: 10.1103/PhysRevLett.64.1196.
- J. T. Overvelde, T. A. de Jong, Y. Shevchenko, S. A. Becerra, G. M. Whitesides, J. C. Weaver, C. Hoberman, and K. Bertoldi. A three-dimensional actuated origami-inspired transformable metamaterial with multiple degrees of freedom. *Nature Communications*, 7(1):10929, dec 2016. ISSN 2041-1723. doi: 10.1038/ncomms10929.

- J. T. B. Overvelde, J. C. Weaver, C. Hoberman, and K. Bertoldi. Rational design of reconfigurable prismatic architected materials. *Nature*, 541(7637):347–352, jan 2017. ISSN 0028-0836. doi: 10.1038/nature20824.
- L. Papadopoulos, M. A. Porter, K. E. Daniels, and D. S. Bassett. Network analysis of particles and grains. *Journal of Complex Networks*, 6(4):485–565, 2018.
- H. Park, A. Niida, S. Miyano, and S. Imoto. Sparse overlapping group lasso for integrative multi-omics analysis. *J Comput Biol*, 22(2):73–84, 2015.
- F. Pasqualetti, S. Zampieri, and F. Bullo. Controllability metrics, limitations and algorithms for complex networks. *IEEE Transactions on Control of Network Systems*, 1(1):40–52, 2014.
- S. N. Patek, B. N. Nowroozi, J. E. Baio, R. L. Caldwell, and A. P. Summers. Linkage mechanics and power amplification of the Mantis Shrimp’s strike. *Journal of Experimental Biology*, 210(20):3677–3688, oct 2007a. ISSN 0022-0949. doi: 10.1242/jeb.006486.
- S. N. Patek, B. N. Nowroozi, J. E. Baio, R. L. Caldwell, and A. P. Summers. Linkage mechanics and power amplification of the mantis shrimp’s strike. *Journal of Experimental Biology*, 210(20):3677–3688, oct 2007b. ISSN 0022-0949. doi: 10.1242/jeb.006486.
- J. Paulose, B. G.-g. Chen, and V. Vitelli. Topological modes bound to dislocations in mechanical metamaterials. *Nature Physics*, 11(2):153–156, jan 2015. ISSN 1745-2473. doi: 10.1038/nphys3185.
- S. Pellegrino. *Deployable Structures*, volume 412. Springer-Verlag Wien, 1 edition, 2001. ISBN 978-3-7091-2584-7. doi: 10.1007/978-3-7091-2584-7.
- B. E. Pfeiffer and D. J. Foster. Hippocampal place-cell sequences depict future paths to remembered goals. *Nature*, 497(7447):74–79, may 2013. ISSN 0028-0836. doi: 10.1038/nature12112.
- T. Pfeil, A. Grubl, S. Jeltsch, E. Muller, P. Muller, M. A. Petrovici, M. Schmuker, D. Bruderle, J. Schemmel, and K. Meier. Six networks on a universal neuromorphic computing substrate. *Front Neurosci*, 7:11, 2013.
- R. C. Picu. Mechanics of random fiber networks - a review. *Soft Matter*, 7:6768–6785, 2011.
- R. Potestio, F. Pontiggia, and C. Micheletti. Coarse-grained description of protein internal dynamics: an optimal strategy for decomposing proteins in rigid subunits. *Biophysical journal*, 96(12):4993–5002, 2009.
- L. Puig, A. Barton, and N. Rando. A review on large deployable structures for astrophysics missions. *Acta Astronautica*, 67(1-2):12–26, jul 2010. ISSN 00945765. doi: 10.1016/j.actaastro.2010.02.021.

- J. Qiao, F. Li, H. Han, and W. Li. Growing Echo-State Network With Multiple Subreservoirs. *IEEE Transactions on Neural Networks and Learning Systems*, 28(2):391–404, feb 2017. ISSN 2162-237X. doi: 10.1109/TNNLS.2016.2514275.
- J. Qin and A. R. Wheeler. Maze exploration and learning in *c. elegans*. *Lab on a Chip*, 7(2):186–192, 2007.
- A. Rafsanjani, L. Jin, B. Deng, and K. Bertoldi. Propagation of pop ups in kirigami shells. *Proceedings of the National Academy of Sciences*, 116(17):8200–8205, apr 2019. ISSN 0027-8424. doi: 10.1073/pnas.1817763116.
- F. Randi and A. M. Leifer. Measuring and modeling whole-brain neural dynamics in *caenorhabditis elegans*. *Current Opinion in Neurobiology*, 65:167–175, 2020.
- D. C. Rapaport and D. C. R. Rapaport. *The art of molecular dynamics simulation*. Cambridge university press, 2004.
- D. Richard, M. Ozawa, S. Patinet, E. Stanifer, B. Shang, S. Ridout, B. Xu, G. Zhang, P. Morse, J.-L. Barrat, et al. Predicting plasticity in disordered solids from structural indicators. *Physical Review Materials*, 4(11):113609, 2020.
- D. Z. Rocklin, S. Zhou, K. Sun, and X. Mao. Transformable topological mechanical metamaterials. *Nature communications*, 8(1):1–9, 2017.
- J. W. Rocks, N. Pashine, I. Bischofberger, C. P. Goodrich, A. J. Liu, and S. R. Nagel. Designing allostery-inspired response in mechanical networks. *Proceedings of the National Academy of Sciences*, 114(10):2520–2525, mar 2017a. ISSN 0027-8424. doi: 10.1073/pnas.1612139114.
- J. W. Rocks, N. Pashine, I. Bischofberger, C. P. Goodrich, A. J. Liu, and S. R. Nagel. Designing allostery-inspired response in mechanical networks. *Proceedings of the National Academy of Sciences*, 114(10):2520–2525, 2017b.
- J. W. Rocks, H. Ronellenfitsch, A. J. Liu, S. R. Nagel, and E. Katifori. Limits of multifunctionality in tunable networks. *Proceedings of the National Academy of Sciences*, 116(7):2506–2511, 2019.
- M. Rubinov, O. Sporns, J. P. Thivierge, and M. Breakspear. Neurobiologically realistic determinants of self-organized criticality in networks of spiking neurons. *PLoS Comput Biol*, 7(6):e1002038, 2011.
- M. Rubinov, R. J. Ypma, C. Watson, and E. T. Bullmore. Wiring cost and topological participation of the mouse brain connectome. *Proc Natl Acad Sci U S A*, 112(32):10032–10037, 2015.
- W. J. Rugh. *Nonlinear system theory*. Johns Hopkins University Press Baltimore, MD, 1981.

- N. F. Rulkov, M. M. Sushchik, L. S. Tsimring, and H. D. Abarbanel. Generalized synchronization of chaos in directionally coupled chaotic systems. *Physical Review E*, 51(2):980, 1995a.
- N. F. Rulkov, M. M. Sushchik, L. S. Tsimring, and H. D. I. Abarbanel. Generalized synchronization of chaos in directionally coupled chaotic systems. *Physical Review E*, 51(2):980–994, feb 1995b. ISSN 1063-651X. doi: 10.1103/PhysRevE.51.980.
- D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.
- J. Ruths and D. Ruths. Control profiles of complex networks. *Science*, 343(6177):1373–1376, 2014.
- T. N. Sainath, B. Kingsbury, G. Saon, H. Soltau, A.-r. Mohamed, G. Dahl, and B. Ramabhadran. Deep Convolutional Neural Networks for Large-scale Speech Tasks. *Neural Networks*, 64:39–48, apr 2015. ISSN 08936080. doi: 10.1016/j.neunet.2014.08.005.
- J. D. Sander and J. K. Joung. CRISPR-Cas systems for editing, regulating and targeting genomes. *Nature biotechnology*, 32(4):347–55, 2014. ISSN 1546-1696. doi: 10.1038/nbt.2842.
- R. A. Santiago. Context discerning multifunction networks: Reformulating fixed weight neural networks. In *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No. 04CH37541)*, volume 1, pages 189–194. IEEE, 2004.
- K. Sato and R. Tanaka. Solitons in one-dimensional mechanical linkage. *Physical Review E*, 98(1):013001, jul 2018. ISSN 2470-0045. doi: 10.1103/PhysRevE.98.013001.
- A. M. Schäfer and H. G. Zimmermann. Recurrent neural networks are universal approximators. In *International Conference on Artificial Neural Networks*, pages 632–640. Springer, 2006.
- M. T. Schaub, N. O’Clery, Y. N. Billeh, J.-C. Delvenne, R. Lambiotte, and M. Barahona. Graph partitions and cluster synchronization in networks of oscillators. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 26(9):094821, 2016.
- B. H. Scheid, A. Ashourvan, J. Stiso, K. A. Davis, F. Mikhail, F. Pasqualetti, B. Litt, and D. S. Bassett. Time-evolving controllability of effective connectivity networks during seizure progression. *Proceedings of the National Academy of Sciences*, 118(5), 2021.
- P. J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of fluid mechanics*, 656:5–28, 2010.
- N. Schweighofer and K. Doya. Meta-learning in reinforcement learning. *Neural Networks*, 16(1):5–9, 2003.

- H. S. Seung. Learning continuous attractors in recurrent networks. In *Advances in Neural Information Processing Systems*, pages 654–660. MIT Press, 1998.
- A. N. Sharkovskii. Coexistence of cycles of a continuous map of the line into itself. *International Journal of Bifurcation and Chaos*, 05(05):1263–1273, 1995. doi: 10.1142/S0218127495000934.
- W. L. Shew, W. P. Clawson, J. Pobst, Y. Karimipour, N. C. Wright, and R. Wessel. Adaptation to sensory input tunes visual cortex to criticality. *Nature Physics*, 11:659–663, 2015.
- F. Shi, S. Wang, M. G. Forest, and P. J. Mucha. Network-based assessments of percolation-induced current distributions in sheared rod macromolecular dispersions. *Multiscale Modeling and Simulation*, 12:249–264, 2014.
- C. T. Shih, O. Sporns, S. L. Yuan, T. S. Su, Y. J. Lin, C. C. Chuang, T. Y. Wang, C. C. Lo, R. J. Greenspan, and A. S. Chiang. Connectomics-based analysis of information flow in the *Drosophila* brain. *Current Biology*, 25(10):1249–1258, 2015.
- J. M. Shine, O. Koyejo, and R. A. Poldrack. Temporal metastates are associated with differential patterns of time-resolved connectivity, network topology, and attention. *Proc Natl Acad Sci U S A*, 113(35):9888–9891, 2016.
- L. E. Silbert. Jamming of frictional spheres and random loose packing. *Soft Matter*, 6(13):2918–2924, 2010.
- J. L. Silverberg, A. A. Evans, L. McLeod, R. C. Hayward, T. Hull, C. D. Santangelo, and I. Cohen. Using origami design principles to fold reprogrammable mechanical metamaterials. *Science*, 345(6197):647–650, aug 2014a. ISSN 0036-8075. doi: 10.1126/science.1252876.
- J. L. Silverberg, A. A. Evans, L. McLeod, R. C. Hayward, T. Hull, C. D. Santangelo, and I. Cohen. Using origami design principles to fold reprogrammable mechanical metamaterials. *science*, 345(6197):647–650, 2014b.
- H. Simon. The architecture of complexity. *Proceedings of the American Philosophical Society*, 10(6):467–482, 1962.
- A. Sizemore, C. Giusti, and D. S. Bassett. Classification of weighted networks through mesoscale homological features. *Journal of Complex Networks*, Published online August 4, 2016, 2016.
- A. E. Sizemore, C. Giusti, A. Kahn, J. M. Vettel, R. F. Betzel, and D. S. Bassett. Cliques and cavities in the human connectome. *Journal of computational neuroscience*, 44(1):115–145, 2018.
- R. Słowiak. Inverses and Determinants of Toeplitz-Hessenberg Matrices. *Taiwanese Journal of Mathematics*, 22(4):901–908, jun 2018. ISSN 1027-5487. doi: 10.11650/tjm/180103.



- A. Sofla, S. Meguid, K. Tan, and W. Yeo. Shape morphing of aircraft wing: Status and challenges. *Materials & Design*, 31(3):1284–1292, mar 2010. ISSN 02613069. doi: 10.1016/j.matdes.2009.09.011.
- O. Sporns and R. F. Betzel. Modular brain networks. *Annu Rev Psychol*, 67:613–640, 2016.
- O. Sporns, R. Kötter, and K. J. Friston. Motifs in brain networks. *PLoS biology*, 2(11):e369, 2004.
- J. C. Sprott and A. Xiong. Classifying and quantifying basins of attraction. *Chaos*, 25(8), 2015. ISSN 10541500. doi: 10.1063/1.4927643.
- M. Stern, V. Jayaram, and A. Murugan. Shaping the topology of folding pathways in mechanical systems. *Nature Communications*, 9(1):4303, dec 2018. ISSN 2041-1723. doi: 10.1038/s41467-018-06720-1.
- M. Stern, C. Arinze, L. Perez, S. E. Palmer, and A. Murugan. Supervised learning through physical changes in a mechanical system. *Proceedings of the National Academy of Sciences*, 117(26):14843–14850, 2020.
- J. Stiso, A. N. Khambhati, T. Menara, A. E. Kahn, J. M. Stein, S. R. Das, R. Gorniak, J. Tracy, B. Litt, K. A. Davis, et al. White matter network architecture guides direct electrical stimulation through optimal state transitions. *Cell reports*, 28(10):2554–2566, 2019.
- S. H. Strogatz. *Nonlinear Dynamics and Chaos*. Perseus Books, 1 edition, 1994. ISBN 0738204536.
- S. H. Strogatz. *Nonlinear dynamics and chaos*. CRC Press, may 2018. ISBN 9780429492563. doi: 10.1201/9780429492563.
- J. U. Surjadi, L. Gao, H. Du, X. Li, X. Xiong, N. X. Fang, and Y. Lu. Mechanical metamaterials and their engineering applications. *Advanced Engineering Materials*, 21(3):1800864, mar 2019a. ISSN 1438-1656. doi: 10.1002/adem.201800864.
- J. U. Surjadi, L. Gao, H. Du, X. Li, X. Xiong, N. X. Fang, and Y. Lu. Mechanical metamaterials and their engineering applications. *Advanced Engineering Materials*, 21(3):1800864, 2019b.
- D. Sussillo and L. Abbott. Generating Coherent Patterns of Activity from Chaotic Neural Networks. *Neuron*, 63(4):544–557, aug 2009. ISSN 08966273. doi: 10.1016/j.neuron.2009.07.018.
- R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- A. Tacchetti, L. Isik, and T. A. Poggio. Invariant Recognition Shapes Neural Representations of Visual Input. *Annual Review of Vision Science*, 4(1):403–422, sep 2018. ISSN 2374-4642. doi: 10.1146/annurev-vision-091517-034103.

- I. Y. Tyukin, D. Prokhorov, and C. Van Leeuwen. Adaptive classification of temporal signals in fixed-weight recurrent neural networks: An existence proof. *Neural Computation*, 20(10):2564–2596, 2008.
- N. Tzourio-Mazoyer, B. Landeau, D. Papathanassiou, F. Crivello, O. Etard, N. Delcroix, B. Mazoyer, and M. Joliot. Automated anatomical labeling of activations in SPM using a macroscopic anatomical parcellation of the MNI MRI single-subject brain. *NeuroImage*, 15(1):273–289, Jan. 2002.
- M. Valencia, M. Pastor, M. Fernández-Seara, J. Artieda, J. Martinerie, and M. Chavez. Complex modular structure of large-scale brain networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 19(2):023119, 2009.
- H. J. van der Horn, J. G. Kok, M. E. de Koning, M. E. Scheenen, A. Leemans, J. M. Spikman, and J. van der Naalt. Altered wiring of the human structural connectome in adults with mild traumatic brain injury. *J Neurotrauma*, page Epub ahead of print, 2016.
- M. F. J. Vermeulen, A. Bose, C. Storm, and W. G. Ellenbroek. Geometry and the onset of rigidity in a disordered network. *Phys. Rev. E*, 96(5):053003, 2017.
- J. von Neumann. First draft of a report on the EDVAC. *IEEE Annals of the History of Computing*, 15(4):27–75, 1993. ISSN 1058-6180. doi: 10.1109/85.238389.
- J. Wang, D. Narain, E. A. Hosseini, and M. Jazayeri. Flexible timing by temporal scaling of cortical responses. *Nature Neuroscience*, 21(1):102–110, jan 2018. ISSN 1097-6256. doi: 10.1038/s41593-017-0028-6.
- D. J. Watts and S. H. Strogatz. Collective dynamics of ‘small-world’ networks. *Nature*, 393(6684):440–2, 1998a. ISSN 0028-0836. doi: 10.1038/30918.
- D. J. Watts and S. H. Strogatz. Collective dynamics of ‘small-world’ networks. *nature*, 393(6684):440–442, 1998b.
- M. Weber, P. D. Maia, and J. N. Kutz. Estimating Memory Deterioration Rates Following Neurodegeneration and Traumatic Brain Injuries in a Hopfield Network Model. *Frontiers in Neuroscience*, 11(NOV), nov 2017. ISSN 1662-453X. doi: 10.3389/fnins.2017.00623.
- G. Wei, Y. Chen, and J. S. Dai. Synthesis, mobility, and multifurcation of deployable polyhedral mechanisms with radially reciprocating motion. *Journal of Mechanical Design*, 136(9):091003, jun 2014. ISSN 1050-0472. doi: 10.1115/1.4027638.
- Z. Wei, R. Sandström, and S. Miyazaki. Shape-memory materials and hybrid composites for smart systems: Part i shape-memory materials. *Journal of materials science*, 33(15):3743–3762, 1998.
- P. J. Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

- J. G. White, E. Southgate, J. N. Thomson, S. Brenner, et al. The structure of the nervous system of the nematode *Caenorhabditis elegans*. *Philos Trans R Soc Lond B Biol Sci*, 314 (1165):1–340, 1986.
- W. Whiteley. Infinitesimal motions of a bipartite framework. *Pacific Journal of Mathematics*, 110(1):233–255, jan 1984. ISSN 0030-8730. doi: 10.2140/pjm.1984.110.233.
- M. O. Williams, I. G. Kevrekidis, and C. W. Rowley. A data-driven approximation of the koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25(6):1307–1346, 2015.
- H. R. Wilson and J. D. Cowan. Excitatory and Inhibitory Interactions in Localized Populations of Model Neurons. *Biophysical Journal*, 12(1):1–24, jan 1972. ISSN 00063495. doi: 10.1016/S0006-3495(72)86068-5.
- S. Wolfram. Cellular automata as models of complexity. *Nature*, 311(5985):419–424, 1984.
- S. Wu, K. Y. M. Wong, C. C. A. Fung, Y. Mi, and W. Zhang. Continuous Attractor Neural Networks: Candidate of a Canonical Model for Neural Information Representation. *F1000Research*, 5:156, feb 2016. ISSN 2046-1402. doi: 10.12688/f1000research.7387.1.
- J. Yang, L. Wang, Y. Wang, and T. Guo. A novel memristive Hopfield neural network with application in associative memory. *Neurocomputing*, 227:142–148, mar 2017. ISSN 09252312. doi: 10.1016/j.neucom.2016.07.065.
- Z. Yang, F. Gao, X. Shi, X. Lin, Z. Gao, Y. Chong, and B. Zhang. Topological acoustics. *Physical Review Letters*, 114(11):114301, mar 2015. ISSN 0031-9007. doi: 10.1103/PhysRevLett.114.114301.
- F.-C. Yeh, V. J. Wedeen, and W.-Y. I. Tseng. Generalized q-sampling imaging. *IEEE transactions on medical imaging*, 2010.
- F.-C. Yeh, T. D. Verstynen, Y. Wang, J. C. Fernández-Miranda, and W.-Y. I. Tseng. Deterministic diffusion fiber tracking improved by quantitative anisotropy. *PloS one*, 8 (11):e80713, 2013.
- A. Yendiki, K. Koldewyn, S. Kakunoori, N. Kanwisher, and B. Fischl. Spurious group differences due to head motion in a diffusion MRI study. *NeuroImage*, 88C:79–90, Nov. 2013.
- K. Yoon, M. A. Buice, C. Barry, R. Hayman, N. Burgess, and I. R. Fiete. Specific evidence of low-dimensional continuous attractor dynamics in grid cells. *Nature Neuroscience*, 16 (8):1077–1084, aug 2013. ISSN 1097-6256. doi: 10.1038/nn.3450.
- Yu Zheng and Wen-Han Qian. Dynamic force distribution in multifingered grasping by decomposition and positive combination. *IEEE Transactions on Robotics*, 21(4):718–726, aug 2005. ISSN 1552-3098. doi: 10.1109/TRO.2005.847609.

- Z. Zhang, Y.-Y. Jiao, and Q.-Q. Sun. Developmental maturation of excitation and inhibition balance in principal neurons across four layers of somatosensory cortex. *Neuroscience*, 174:10–25, feb 2011. ISSN 03064522. doi: 10.1016/j.neuroscience.2010.11.045.
- Z. Zhao, X. Kuang, J. Wu, Q. Zhang, G. H. Paulino, H. J. Qi, and D. Fang. 3D printing of complex origami assemblages for reconfigurable structures. *Soft Matter*, 14(39):8051–8059, 2018. ISSN 1744-683X. doi: 10.1039/C8SM01341A.
- C. Zhou, L. Zemanová, G. Zamora, C. C. Hilgetag, and J. Kurths. Hierarchical organization unveiled by functional connectivity in complex brain networks. *Physical review letters*, 97(23):238103, 2006.
- Y. Zhou, B. G.-g. Chen, N. Upadhyaya, and V. Vitelli. Kink-antikink asymmetry and impurity interactions in topological mechanical chains. *Phys. Rev. E*, 95:022202, Feb 2017. doi: 10.1103/PhysRevE.95.022202.
- B. Zhu and Y. Xia. An information-theoretic model for link prediction in complex networks. *Sci Rep*, 5:13707, 2015.
- O. C. Zienkiewicz, R. L. Taylor, P. Nithiarasu, and J. Zhu. *The finite element method*, volume 3. McGraw-hill London, 1977.
- L. Zigoneanu, B.-I. Popa, and S. A. Cummer. Three-dimensional broadband omnidirectional acoustic ground cloak. *Nature Materials*, 13(4):352–355, apr 2014. ISSN 1476-1122. doi: 10.1038/nmat3901.